WHOI 90-39

A SAIL Compatible Three Channel Acoustic Navigation Interrogator

by

Stephen P. Liberatore

Woods Hole Oceanographic Institution
Woods Hole, Massachusetts 02543

September 1990

Technical Report

Funding was provided by the Office of Naval Research under Contract Nos. N00014-82-C-0152 and N00014-85-C-0379.

Approved for Distribution:

Albert J. Williams 3rd

Albert J. Williams 3rd, Chairman
Department of Applied Ocean Physics & Engineering

ABSTRACT

Ocean Acoustic Tomography data are significantly degraded if mooring motion is unknown. An autonomous instrument using employing a solid state data logger designed to track and record mooring motion is described.

Navigation is accomplished by simultaneously interrogating each of three bottom mounted transponders positioned in an equilateral triangle around the mooring's anchor at a range approximately equal to the depth of the tracked instrument. The three round-trip travel times thus obtained, having a resolution of 125uS and a SNR dependent jitter of less than 1.5mS, define a unique instrument position and are recorded along with the time of day and day of year.

The measurement period, the system clock and the program start time are set via a 20mA SAIL. Since the standby power requirement is negligible compared to the battery capacity, the instrument may be programmed months in advance of the deployment.

System endurance varies with the measurement period, however, typical programs permit navigation for up to 21 months at 12 points per day.

Upon recovery, the navigator data may be down-loaded via SAIL directly to the storage medium of a suitable computer.

Keywords: Position finding; Recording systems; Computer programs/interrogators; Data processing equipment. Autonomous navigation. (EDC)

Table of Contents

1.0 GENERAL DESCRIPTION

1.1 Introduction

The requirement to spatially track acoustic transceivers moored as part of an Ocean Acoustic Tomography experiment has led the Woods Hole Oceanographic Institution and Benthos Inc. of Falmouth, Ma, to develop an acoustic mooring navigation system.

The electronics module designed at W.H.O.I. and described in this manual is used with the BENTHOS model (ES) 210-TCSSA acoustic transceiver. Together they form a Mooring Motion Monitoring Module (QUAD M) Interrogator.

This document serves as a system hardware reference manual for the technical, but uninitiated user. It references other hardware manuals where appropriate and provides system-oriented information unavailable elsewhere. A copy of the interrogator control program (PNAVLGR) is included as an addendum to this manual.

1.2 System Components

Tracking is accomplished by measuring round-trip travel time from the interrogator to three transponders. The transponders are moored about three meters above the ocean floor and approximately one water depth away from the mooring anchor.

Figure 1 is a block diagram of a mooring equipped to monitor the motion of an instrument mounted near a sub-surface float. "A", "B", and "C", are bottom-mounted acoustic transponders, either Benthos model 210-TR17A-GF which are recoverable or model XT-6000 which are not. The interrogator is mounted as near as practical to the instrument tracked. The frequencies depicted are those which were originally employed. To remain compatible with as many tomography instruments as possible, the 13.5kHz channel has been retuned to 12.0kHz.

The interrogator pings to all three transponders simultaneously at a predetermined time and at a predetermined rate. The time required to receive a response from each transponder, along with the time of day and date, are stored in CMOS static RAM.

The operating parameters are set via the Serial ASCII Instrumentation Loop (SAIL). Pre-deployment checks and data retrieval are also accomplished over the SAIL. A formal description of the SAIL standard is presented in U.N.O.L.S. Ref. TAC-81-1 Aug. 1981, "Serial ASCII Instrumentation Loop (SAIL)" or IEEE standard 997-1985.

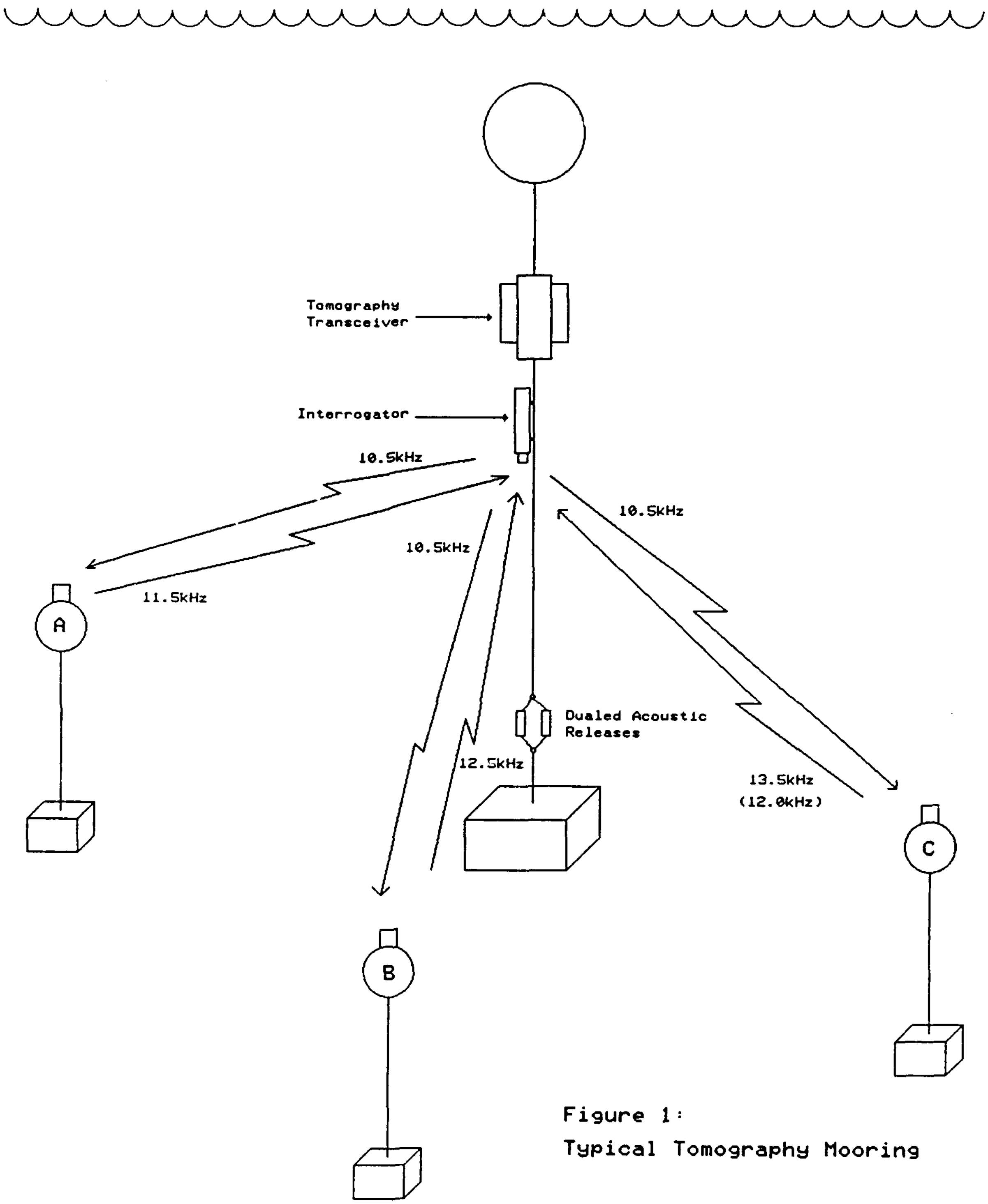


Figure 1:
Typical Tomography Mooring

2.0 SPECIFICATIONS

2.1 Interrogator

The transceiver specifications, except the electrical power source and operating life, are as listed in the Benthos operating manual for the (ES)210-TCSSA. These two exceptions are the result of replacing a MICRO tape recorder and its associated control electronics with a solid state memory and a power-switched, microprocessor-based controller. The transceiver configured in this manner will henceforth be referred to as an interrogator.

2.2 Power

Twenty-one 1.5 volt "D" size alkaline cells supply power for the interrogator. The DURACELL B1300-T2, with spot welded solder tabs on both terminals is the preferred cell.

The cells are configured as follows: Two diode-isolated parallel strings, each consisting of 9 cells are wired in series yielding 12 volts, then 3 cells are wired in series with the 12 volt stack to yield 16 volts. The battery thus formed is tapped at 12 volts to power the acoustic receiver and the digital electronics, while the 16 volt tap supplies the pinger's power amplifier.

De-rating for temperature and storage, and assuming an average cell voltage of 1 volt, each cell will yield approximately 10 watt hours. The above stack is therefore rated at 210 watt hours.

Making one measurement per hour, the interrogator requires fewer than 0.0045 watt hours. This yields an operating life in excess of 5 years, which exceeds the nominal self discharge time of an alkaline cell. It is however, recommended that the battery be replaced before each deployment.

2.3 Schedule

A measurement may be made as often as every three minutes, or as seldom as once every 999 minutes. The time-of-day clock must be set to the nearest whole minute. Assuming that the clock's oscillator was adjusted to 32.768kHz with the interrogator at the same temperature encountered while deployed, its time will be accurate to within +/- 5 minutes after 365 days, i.e., the clock will lose or gain about 1 second per day. The start of a measurement sequence may be scheduled on any whole minute of the year. Leap years are not accounted for so the clock will reset to day 1 on day 366 of a leap year. **Note:** Interrogator S/N 005 has an alternate program allowing it to make measurements as often as every 3 seconds or as seldom as every 999 seconds. This system is typically employed as a recording acoustic range finder for towed instruments.

2.4 Data Format

The 60K RAM (Random Access Memory) allows space for 7648 measurements which, at 12 measurements per day, yields a system endurance in excess of twenty months. After the 7649th measurement, which will be made but not stored, the system will enter the "idle" mode, and no further measurements will be made.

Each measurement consists of a 16 bit time-of-day word, and three 16 bit two-way travel time words. The time of day is recorded with a resolution of one hour. An LSB or travel time is equal to 250 uS. Measurement data, stored beginning at RAM address 1000H, are ordered as follows:

Time of day, Travel time A, Travel time B, and Travel time C.

The time of day is encoded as follows:

BIT #	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UNITS	HD	HD	TD	TD	TD	TD	UD	UD	UD	UD	TH	TH	UH	UH	UH	UH
WEIGHT	2	1	8	4	2	1	8	4	2	1	2	1	8	4	2	1

Where HD is hundreds of days, TD is tens of days, UD is units of days, TH is tens of hours, and UH is units of hours.

As an example, a time code word of 11D6H would convert to day 047 hour 16 as follows:

```
  1    1    D    6
00 0100 0111 01 0110
HD  TD   UD  TH  UH
 0   4   7    1   6
```

2.5 Transponder

The transponder specifications may be found in BENTHOS report 0-210-TR17A-GF or the XT-6000 Technical Manual.

3.0 OPERATION

3.1 Power On / Reset

Following the instructions in Benthos manual 0-210-TCSSA, section 2.1, remove the electronics from the pressure housing. Position the electronics with the back-plane wiring facing away from you and with the transducer on your left. Locate the power switch near the transducer end of the instrument and ensure that it is in the "on" position. Locate the reset pins on the opposite end of the instrument and short them together for at least five seconds. This will reset the digital electronics and start the microprocessor.

3.2 Connect to SAIL

Connect to the SAIL via the banana jacks on the controller electronics card. Insure that the loop is closed and connect a terminal to the SAIL / RS-232 converter. Set the terminal for seven data bits, even parity, 1 stop bit, and 300 baud.

3.3 Monitor Current

Connect a digital voltmeter between test points 1 and 2 which are located on either side of R1 on the System Control card. The meter will read total system current scaled at 100uA/mV.

Once the SAIL loop is closed and a full minute has elapsed, the voltmeter will read between 60 and 80 mV. If less than a minute has elapsed the reading may be between .3 and .6 mV. **Wait for the higher reading** which indicates that the processor is awake and ready for SAIL control.

Note: Most of the interrogators are now equipped with a LED to monitor the switched power. With these instruments there is no need to monitor the voltage across R1. Simply wait for the LED to light before attempting to address the interrogator.

3.4 Address

Once the microprocessor has detected the presence of a closed SAIL and applied power to the rest of the system, the instrument may be addressed by typing **#In** where n is the interrogator's serial number. A correctly addressed instrument will respond with:

```
    In READY
           :
                EXAMPLE            #I3  <--- You type this line
                                   I3 READY <--- Interrogator
                                   :        <--- reply
```

The ":" in the above example is the system prompt and signifies that the interrogator is awaiting commands. Type an H and the interrogator will print a list of the available commands.

EXAMPLE

```
: H

 INTERROGATOR PROGRAM     Ver. 1.1   Jan.  1985


 SYSTEM COMMANDS

 !Maaaa dddd         LOAD MEMORY
 ?M                  DISPLAY MEMORY
 ?Paaaa              RUN PROGRAM
 ?C                  CALCULATE CRC
  M                  MOVE MEMORY
  R                  TEST RAM
 ?S                  DISPLAY SCHEDULE
 !SCHEDULE           PROGRAM SCHEDULE
 !TIME               SET CLOCK
 ?T                  DISPLAY TIME
 !LOCK               PROTECT MEMORY
 !UNLOCK             UNPROTECT MEMORY
 !IDLE               INHIBIT SCHEDULER
 !PING               TRANSMIT A 10mS PULSE
```

3.5 Entering Commands

To initiate a command, simply type it exactly as it is listed in the "HELP" file. An error message will be printed in response to an unrecognized command. Usually this message will be followed by the "prompt", at which time you may try re-entering the command. **NOTE: Commands are NOT terminated with a "Carriage**

Return", but ALL numeric entries in response to system prompts MUST be terminated with a "Space".

3.6 Correcting Errors

Numeric entries are expected to be a certain number of digits in length. For example, when entering the start hour, a two digit figure is expected; but when entering the measurement interval, a three digit figure is expected. **Only the last n digits typed prior to a "Space" are entered** (**n** is the number of digits expected). Because of this, typing errors may be corrected by simply typing the correct figure immediately after the error. For example, when entering the measurement interval, if you mistakenly type 20 when what you really wanted was 120, the corrected entry would look like this: 20120. Similarly, an hour entry of 2314234121 would be accepted as hour 21.

3.7 PROM Test

Test the system program memory by typing ?C and answering the questions with **0** over **800**, and **800** over **800**. Verify the correct response by comparing the calculated CRC with the values recorded on the PROMS, IC 4 and 5.

```
EXAMPLE      : ?CRC From 000 Over 800  = 994C
             : ?CRC From 800 Over 800  = EF9A
             :
```

3.8 RAM Test

Test the system RAM by typing **!UNLOCK.** The system will respond with OK. Then type an **R.** The system will respond by typing a cosmetic "am" and the words "Test From". You answer with **1000**, and the system will then type Over, to which you answer **F000**. A RAM test over this much memory requires about one minute and seven seconds. After each successful pass, the system will type a *. Ten such passes would indicate good memory. Reset and address the system as in **3.1 and 3.4** respectively.

```
EXAMPLE        : !UNLOCK  OK
               : Ram Test From 1000 Over F000   OK (Y/N) ? Y
**********
```

The !UNLOCK command is required since RAM test will overwrite any measurements previously stored. The program will automatically execute the !LOCK command when the RAM test is terminated.

3.9 Clock Set

Set the system clock by typing **!TIME DDD HH MM 00** where DDD is the year day, HH is hours and MM is minutes. Since the interrogator clock has a one minute resolution, seconds must always be entered as 00 and the clock must be started on the minute. When real time is equal to the time entered, type an @. This will start the clock. To verify that the correct time was entered and that the clock is running, re-address the instrument (Section 3.4) and after the prompt, type ?T. The interrogator will respond with the current time plus one minute, wait for the real time to equal the time just printed and, on the mark, printing an @.

```
EXAMPLE          !TIME 123 21 35 00 @
                 #In
                  #In READY
                  : ?T  123 21:36 00 Z...@
                  :
```

3.10 Schedule

Set the operating schedule by typing **!SCHEDULE**. The Interrogator will ask you for Start day, hour, minute, and the measurement interval. Terminate all entries with a **SPACE**. When all parameters have been entered, the interrogator will ask permission before activating the scheduler.

```
EXAMPLE   : !SCHEDULE
          Start on day = 115   Hour = 18   Minute = 30
          Measurement interval,  minutes = 060   OK (Y/N) ? Y
```

3.11 Verify Schedule

Verify that the schedule has been accepted as entered by typing ?S. The interrogator will respond by typing the current time and schedule in addition to the system status (ARMED, not ARMED, or ACTIVE). If the system is ACTIVE, the number of minutes remaining to the next measurement (in HEX) and the current data address pointer will also be shown.

```
     EXAMPLE  : ?S

                    At  115 18:10
              Start on day = 115  hour = 18  minute = 30
              Measurement interval = 060 minutes
              Scheduler is ARMED BUT NOT ACTIVE
```

3.12 Test Pinger

Test the pinger by typing **!PING**. The interrogator will respond by typing OK (Y/N) ? If you next type a **Y** you should hear the transmit pulse.

```
     EXAMPLE   : !PING  OK  (Y/N)  Y
               :
```

3.13 Final Test

Disconnect the SAIL cable and observe the system current immediately drop to some value below 100uA. At the next one minute mark, the current will rise to a level near 7mA and stay at that level for about 70mS. If the interrogator is equipped with a LED, it will dimly flash. These observations indicate that the interrogator is functioning correctly and the instrument may be encased in its pressure housing. Refer to section 2.1 of

BENTHOS manual (ES) 210-TCSSA and, following instructions there, place the electronics within the pressure housing. At this point the interrogator is ready for deployment.

3.14 Data Recovery (fast)

When the instrument is recovered, the data which are stored in RAM may be down loaded at a high baud rate directly to the storage medium of a suitably equipped computer. **Be careful not to interrupt power to the system in any way as this WILL result in lost data.** Proceed as follows:

a. Remove the electronics from the pressure housing (3.1)

b. Connect the SAIL to RS-232 converter box. (3.2)

c. Monitor current, and wait for the high reading. (3.3)

d. Replace the jumper plug located on the control card (P3) with the cable from the 5 VOLT BAUD RATE GENERATOR. Set the baud rate generator for 9600 baud. (see Figure 8.)

e. Connect the auxiliary I/O port of the computer to the RS-232 connector on the SAIL to RS-232 converter.

f. Set this port for 9600 baud, seven data bits, one stop bit, and even parity.

g. Using the computer terminal (and the appropriate communications program) address the interrogator. (3.4)

h. Type ?S to verify that the system is still "ACTIVE", that the clock is still running, and to obtain the data address pointer. Subtract 1000H from the current address pointer, and make note of the result.

i. Type **!IDLE** to inhibit further measurements.

j. Prepare the computer to receive an ASCII data file, and type **?M**. The system will respond by printing From. You respond by typing **1000**. The system will then print Over, and you respond by typing the result of the calculation done in 3.14 (h.) followed by a carriage return.

The interrogator down loads two measurements per line. A full memory (7648 measurements) requires approximately three minutes to down load.

4.0 THEORY OF OPERATION

4.1 Acoustic Electronics

Section 5 of Benthos report O-210-TCSSA explains the operation of the acoustic electronics.

4.2 Power Supply

Refcr to Figure 2, which is a simplified block diagram of the interrogator. The capacitor board, the 5 volt regulator, and the low voltage detector are the only blocks which receive power directly from the battery. The 5 volt regulator supplies power on a continuous basis to two other blocks, the clock, and the 60K CMOS static RAM. All other blocks are powered intermittently.

Refer to Figure 3, which is a schematic drawing of the interrogator power supply. These components are located on the SYSTEM CONTROL PC card. Rl is in series with the 12 volt stack, and is used as a current sense resistor for the entire electronics package. A voltmeter placed across this resistor will display current scaled at 100 uA/mV. The ICL 7663 is a micro-power voltage regulator with over-current sense. The output of this regulator is set to 5.5 volts by adjusting Pl. The 2N3643 is a series pass transistor used to supply surge current during the power-up sequence.

The ICL7665 is a micro-power under voltage detector. Its

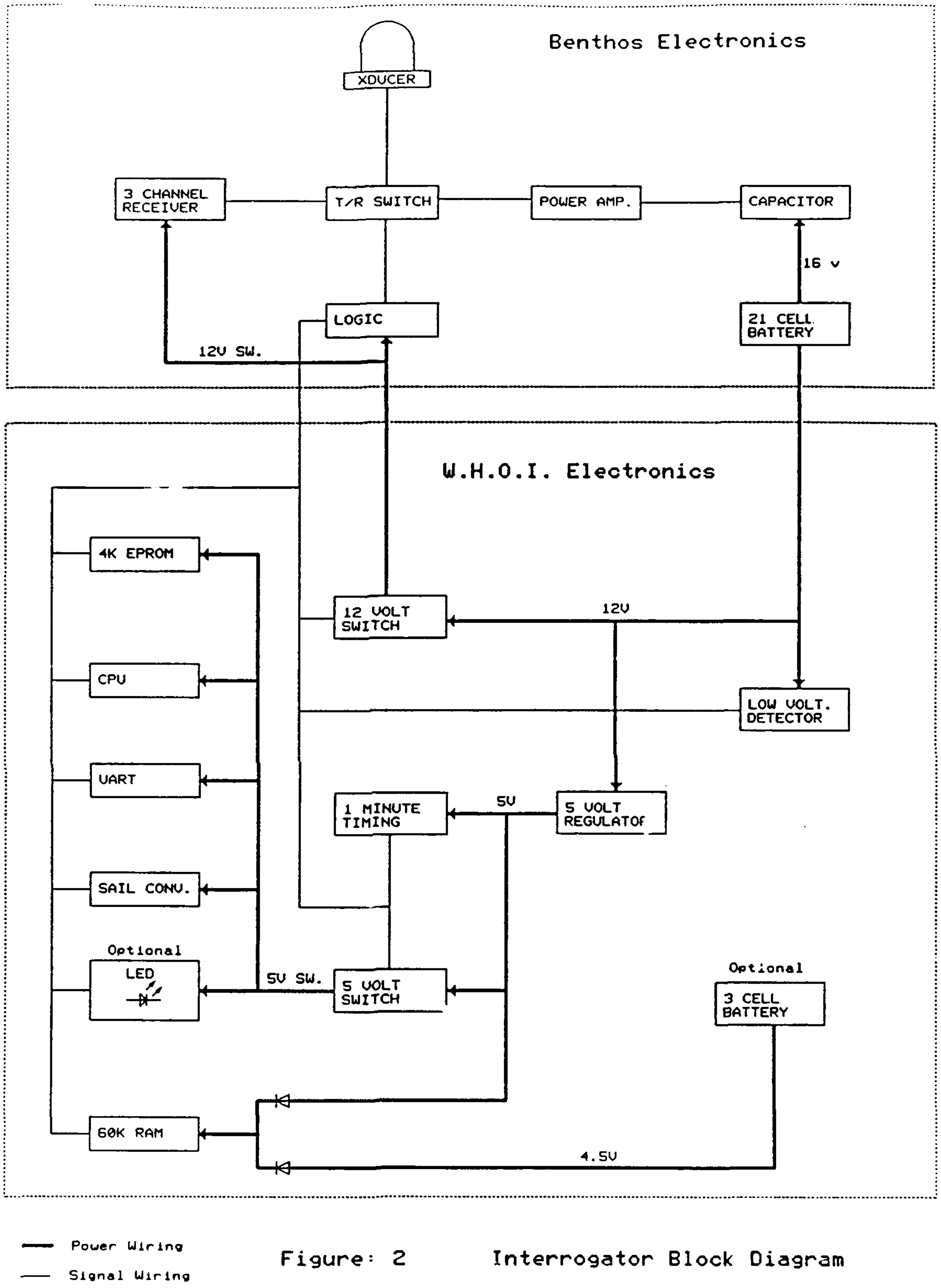


Figure: 2 Interrogator Block Diagram

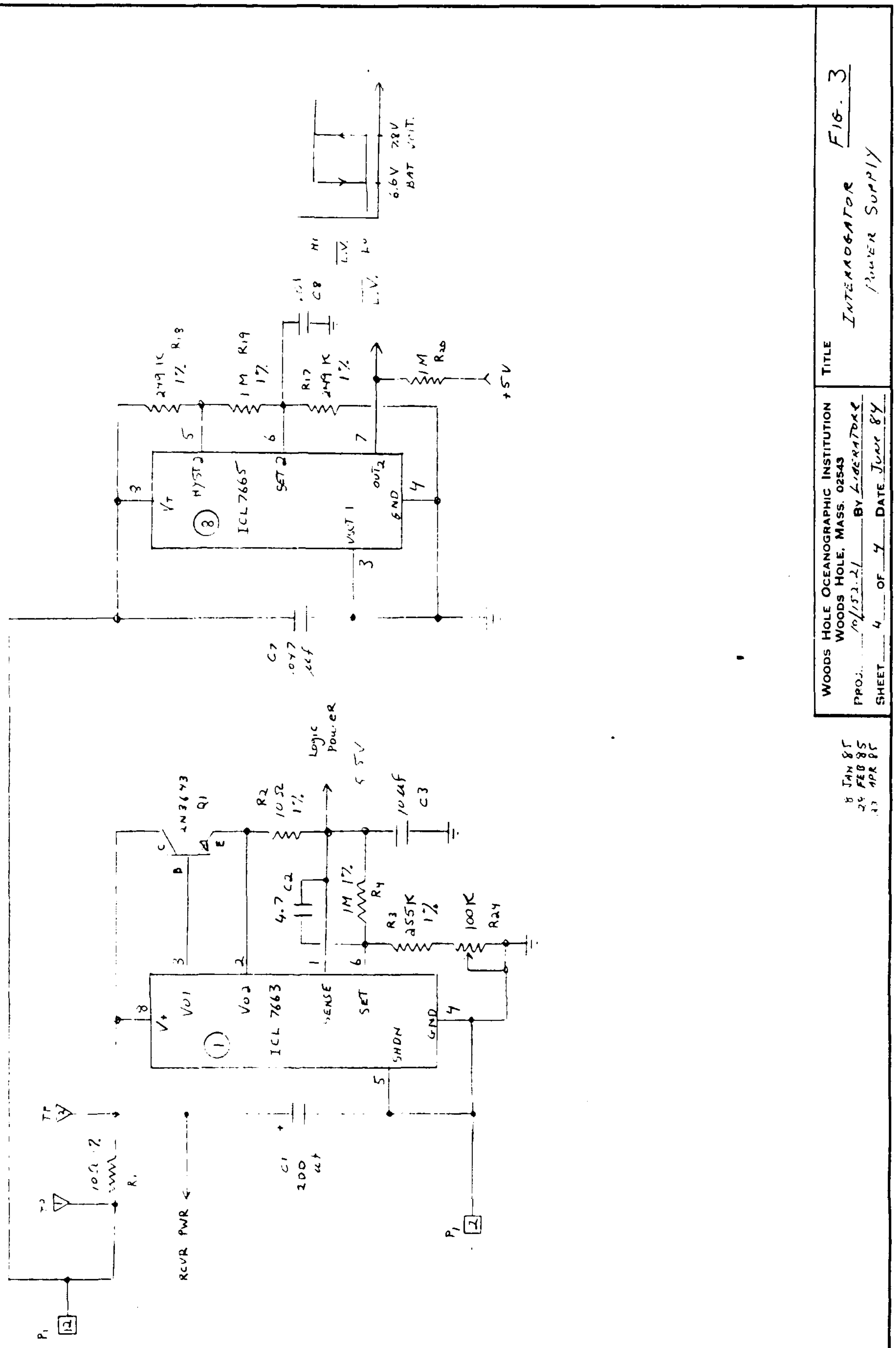


Figure 3: Interrogator Power Supply Schematic

purpose is to monitor the battery and at a preset voltage inhibit further measurements in order to conserve battery power for data retention. When the battery voltage drops below 6.6 volts, LV NOT goes true (logic 0). This will stop a measurement in progress, and inhibit any further measurements from being initiated. LV NOT will remain true until the input voltage on P1-12 rises above 7.8 volts. The 1.2 volt hysteresis prevents the switch from oscillating between true and false, which could occur due to the difference between the open circuit voltage of the battery and the battery voltage while the system is enabled.

4.3 System Control

Refer to Figure 4. This is a schematic of the interrogator system control. These components are located on the same card as the power supply. 5 volt logic power enters through diode D1. This diode drops approximately .5 volts so that VCC and VDD to all components on this card will equal about 5 volts. If this is not the case, check the adjustment of R24.

IC 5,6, and 7 provide a once-per-minute pulse. If the rest of the system is already powered, this pulse simply generates an interrupt for the microprocessor (IC9). If the rest of the system was not already powered, the once-per-minute pulse will clock a HIGH to pin 13 of IC 4. This causes pin 4 of IC 2 to go LOW which enables system memory and turns on Q2.

VDD is applied to the remaining unpowered ICs on this card when Q2 is on. IC 11, which was already powered, now has VDD on input pins 3 and 6. VDD is level shifted via this IC to 12 volts and fed through P1 directly to the BENTHOS electronics.

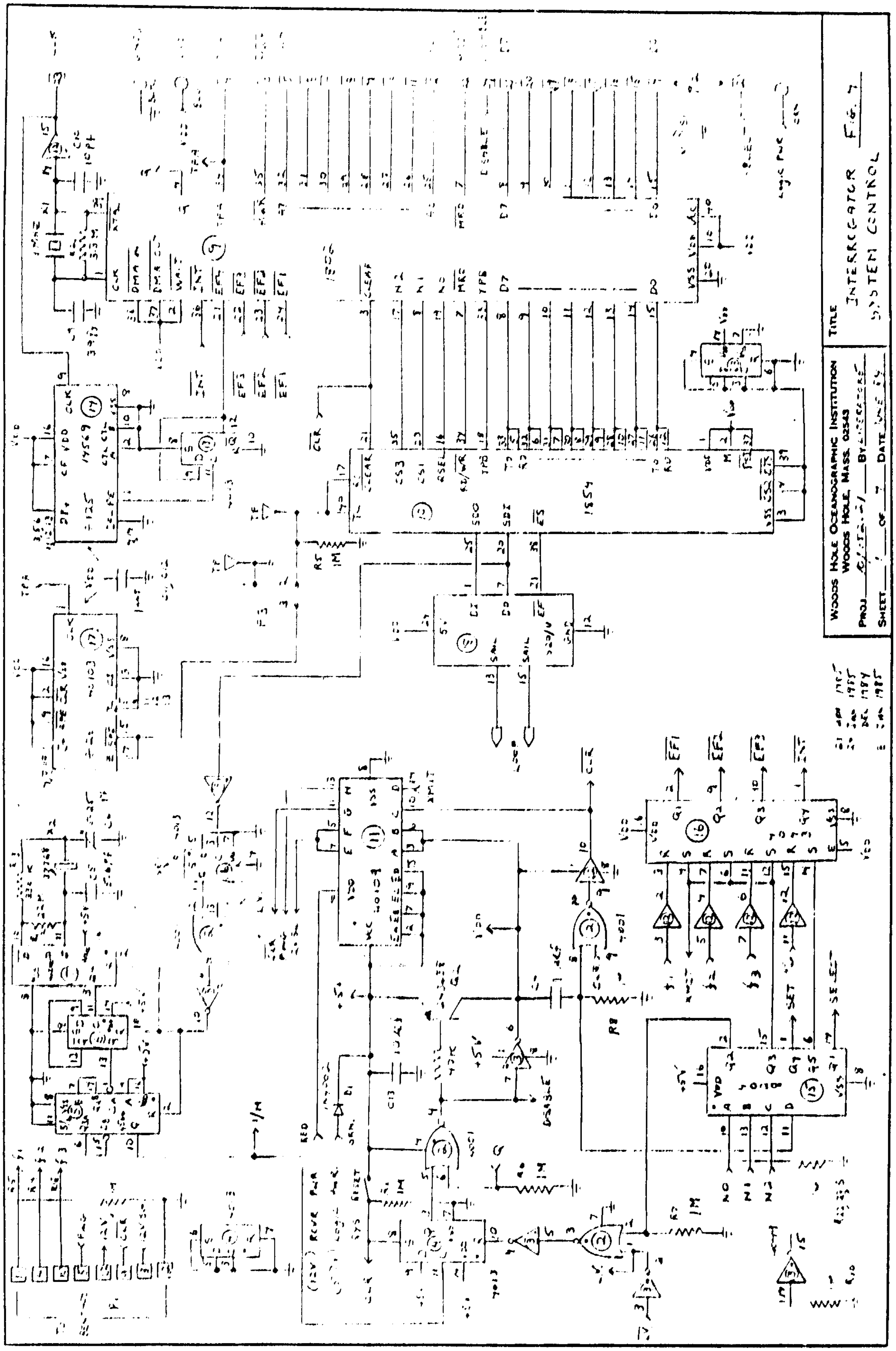
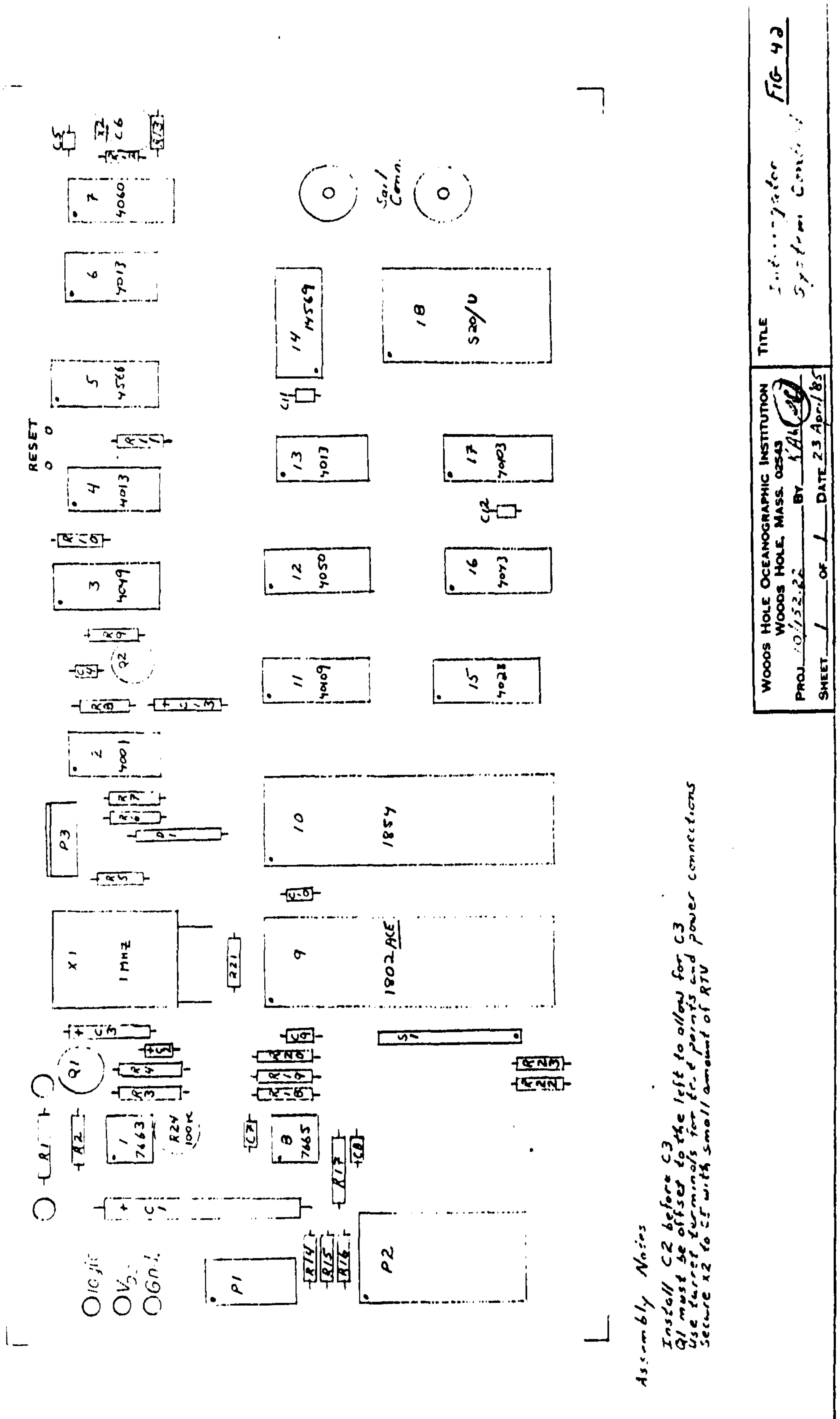


Figure 4: Interrogator System Control Schematic



Assembly Notes
Install C2 before C3
Q1 must be offset to the left to allow for C3
Use turret terminals for test points and power connections
Secure X2 to C5 with small amount of RTV

Figure 4a: Interrogator System Control
Component Location

When VDD first goes high, a reset pulse is generated via C4 charging through R8. The reset pulse is applied directly to pin 11 of IC 15 which inhibits this IC and prevents inadvertent I/O operations. The reset pulse is also inverted via IC 2 and 12. The inverted reset (CLR NOT) is level shifted via IC 11 and routed to the BENTHOS transmitter through P1. This signal, along with a slight modification to the BENTHOS electronics, prevents the transmitter from pinging upon power up. CLR NOT is also connected to IC 9 and 10. IC 9 is the microprocessor, and when CLR NOT goes HIGH, program execution begins at address 0000. The software clock is updated once the program has been initialized, and the UART (IC 10) is examined to determine if the SAIL is open or closed. If the loop is found to be open, a test is made to determine if it is time to begin a measurement cycle. If the loop is closed, interrupts are enabled and take over the function of updating the clock. If the loop is open and it is not time to begin a measurement the microprocessor generates a signal which appears on IC 15 pin 2. This signal is then gated to the reset pin of IC 4 via the OR gate composed of IC 2 and 3. Resetting IC 4 causes a HIGH to appear on pin 4 of IC 2 which will disable the memory select circuits and cause Q2 to turn off. The disable signal is inverted by IC 3, and the LOW thus produced is connected to VDD. Since Q2 is no longer conducting, this LOW will cause VDD to drop rapidly.

NOTE: It is important to remember that the microprocessor reacts to a manual reset in exactly the same fashion that it reacts to the once-per-minute tick. **For this reason, the interrogator clock, which resides only in software, will be advanced one minute with each manual reset, regardless of how much time has actually elapsed.**

IC 14 and 13 divide the 1MHz clock by 250 to produce a 4 kHz square wave which is applied to pin 21 of IC 9. During a measurement sequence, the microprocessor will increment three separate counters on each rising edge of this signal. The action begins immediately after a ping is transmitted, and continues until either all three transponders reply or the counters overflow. The reply detected signals (f1,f2, and f3) from the BENTHOS electronics enter through P1, are level shifted by IC 12, and latched by IC 16. The output of the latch is connected to pins 22, 23, and 24 of the microprocessor; these are three of the flag lines. When the microprocessor detects one of these flags, it stops incrementing the counter associated with that reply channel. The number remaining in the counter represents the two-way travel time. A counter which contains all zeros has overflowed and indicates no reply on that channel.

IC 18 converts the 20 mA SAIL levels to 5 volt CMOS levels for the UART, and provides an output which indicates an open loop. IC 17 divides the TPA clock signal from IC 9 by 26 to provide the 16X clock rate the UART requires to run at 300 baud.

The Q4 output of IC 15 and the DO output of IC 18 synchronize the clock. Once the time has been entered, the microprocessor generates a signal which causes Q4 of IC 15 to go HIGH. This is the SET signal and is applied to the set input of IC 6. Pin 1 of this IC goes HIGH and is gated by the OR gate formed with IC 2 and 3 to the reset inputs of IC 5,6, 7. This stops the clock's oscillator and resets its down counters. The start bit of any character typed over the loop will be inverted by IC 3 and used to clock IC 6. This will remove the reset and allow the clock's oscillator and down counters to operate. If the character was not an "@", the microprocessor will again generate the signal which causes Q4 of IC 15 to go HIGH, and the cycle repeats.

4.4 Memory Control

Refer to Figure 5. This is a schematic of the memory control electronics. These components are located on the 64K memory card.

IC 17 gates the buffered MWR NOT and MRD NOT signals with the DISABLE signal generated on the system control card. This signal will go true just before power is removed from the microprocessor. When disable is true, both XMWR NOT and XMRD NOT are false (logic "1"). XMRD NOT being HIGH holds IC 21 reset. The Q4 output of IC 21 is applied to pin 8 of IC 13; and since pin 9 of this IC is also HIGH, its output, pin 10, is LOW. This is the memory on (or enable memory bus) signal, and when LOW, inhibits all memory operations by de-selecting the memory chips and by turning off the memory bus drivers.

IC 16, 18, and the remaining NAND gates of IC 13 decode the address lines to produce the 8K selects which enable the HM6264 RAM chips on this card. IC 12 decodes the proper address lines to produce the 2K selects which are required by the 27C16 PROM chips, and the HM6116 RAM chips. Since the PROM is power switched, the 2K selects used by these chips are buffered by IC 19. IC 21 is a counter and, with IC 13, is used to truncate the memory cycle and thus conserve power. **It is recommended that the jumper from pin 8 of IC 13 to pin 11 of IC 21 be moved to pin 13 of IC 21, thereby increasing the memory enabled time by 1uS.** This modification, although not essential and causing a slight increase in power consumption, will improve the system's reliability.

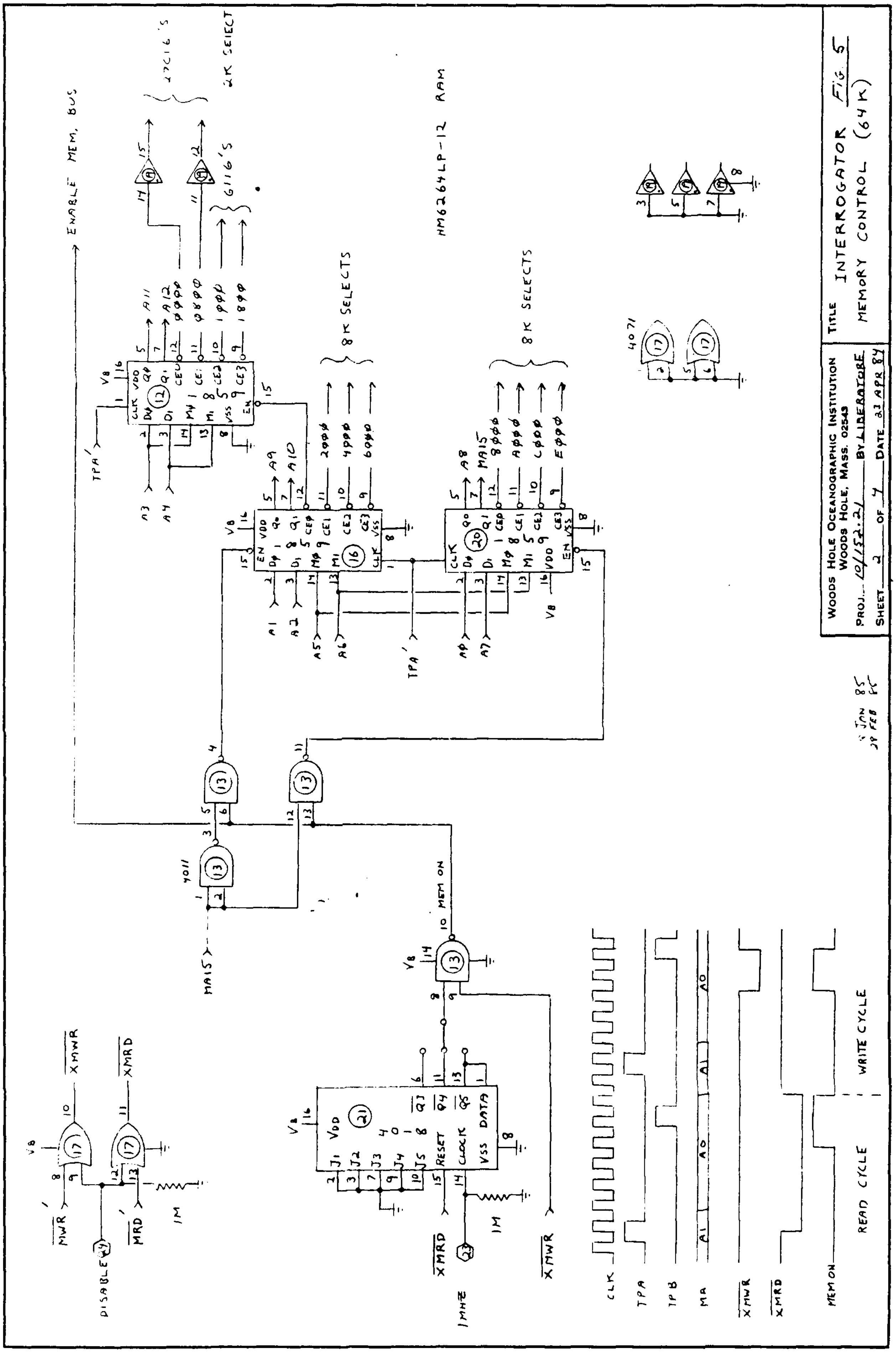


Figure 5: Interrogator Memory Control (64K) Schematic

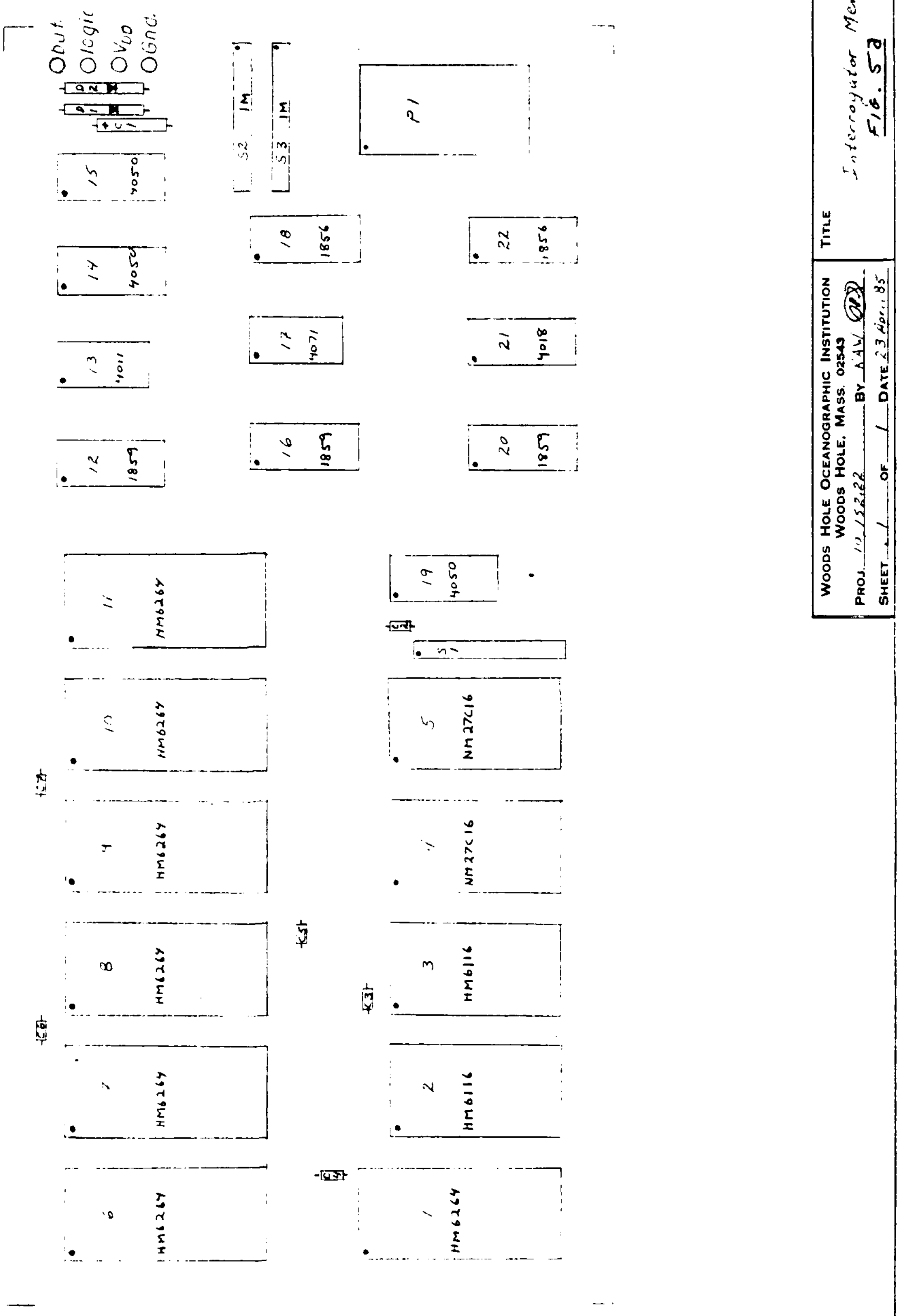


Figure 5a: Interrogator Memory (64K)
Component Location

4.5 64K Memory

Refer to Figure 6. This is a schematic of the system memory. These components are located on the same card as the memory control electronics. A 24 pin ribbon cable connects the memory card to the system control card. The memory is fully buffered by IC 18 and 22 which buffer the data lines and IC 14 and 15 which buffer the address and clock lines. Since IC 4 and 5 are power switched the MRD NOT signal is buffered by IC 19.

Power for this card is supplied via a disconnect through two diodes which ısolate the logic power from the memory back-up battery. The back-up battery is composed of three AAA cells wired in series and, if used, is mounted on the rail over the system control card.

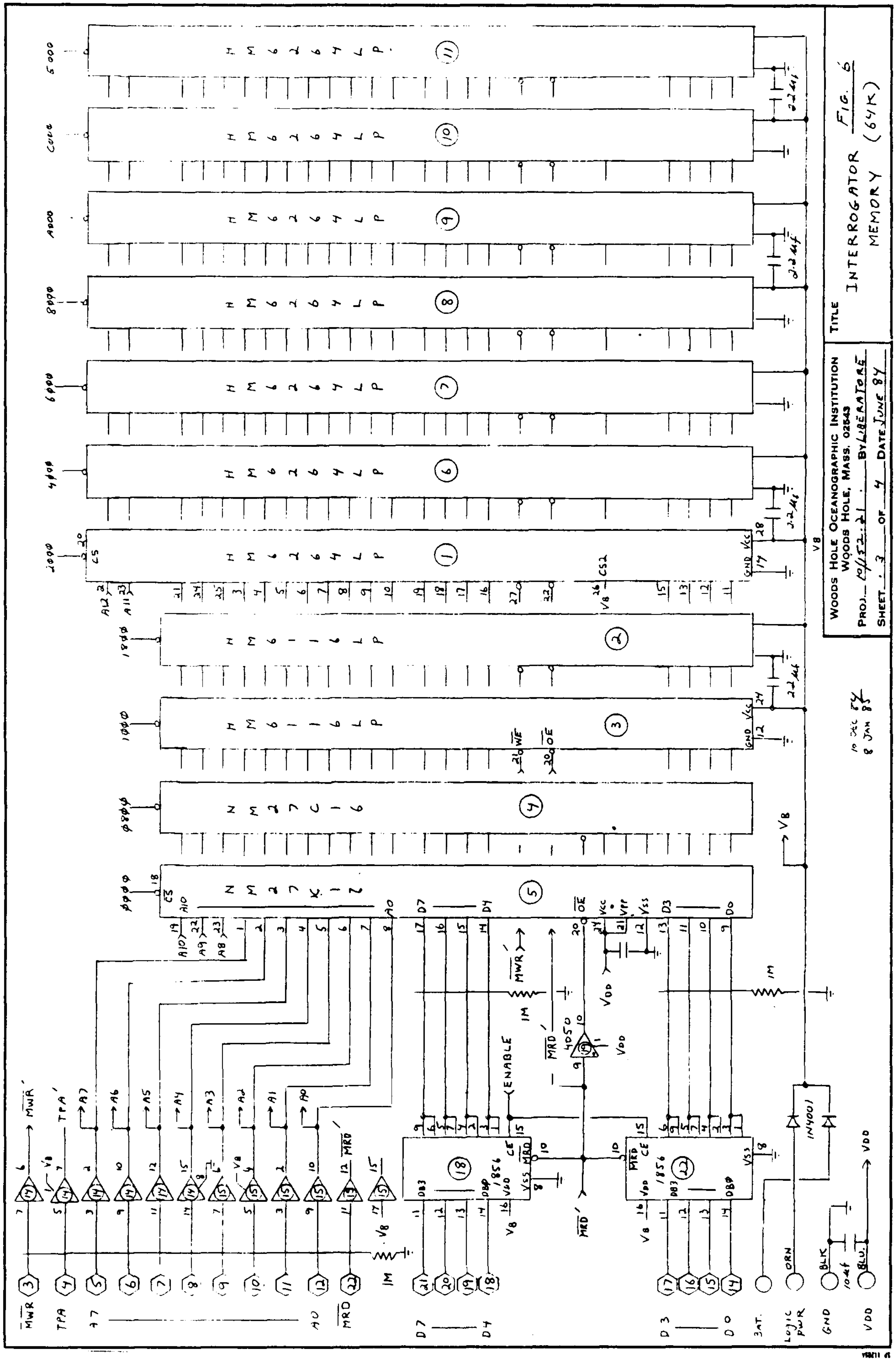


Figure 6: Interrogator Memory (64K) Schematic

5.0 MODIFIED BENTHOS ELECTRONICS

Slight modifications were made to the electronics supplied by BENTHOS. The effects of these modifications are as follows:

a. A six-volt tap from the battery stack is eliminated.

b. Transmitting on every power-up sequence is prevented.

5.1 Logic Board

Refer to BENTHOS drawing B-210-248. This is a schematic for the LOGIC board which must be modified to make provision for a power-up reset pulse. The power-up reset pulse originates on the system control card and inhibits the pinger during the power on cycles which occur at the rate of one per minute. Remove the LOGIC board from the chassis and locate IC 2, a CD4098B. Remove the etch between pins 3, 16, and 13 of IC 2. Connect pin 13 to pin 16 with a short jumper. Connect pin 3 to board I/O pin 10 with another short jumper. Clean the board of flux, and re-coat the patched area with a clear acrylic.

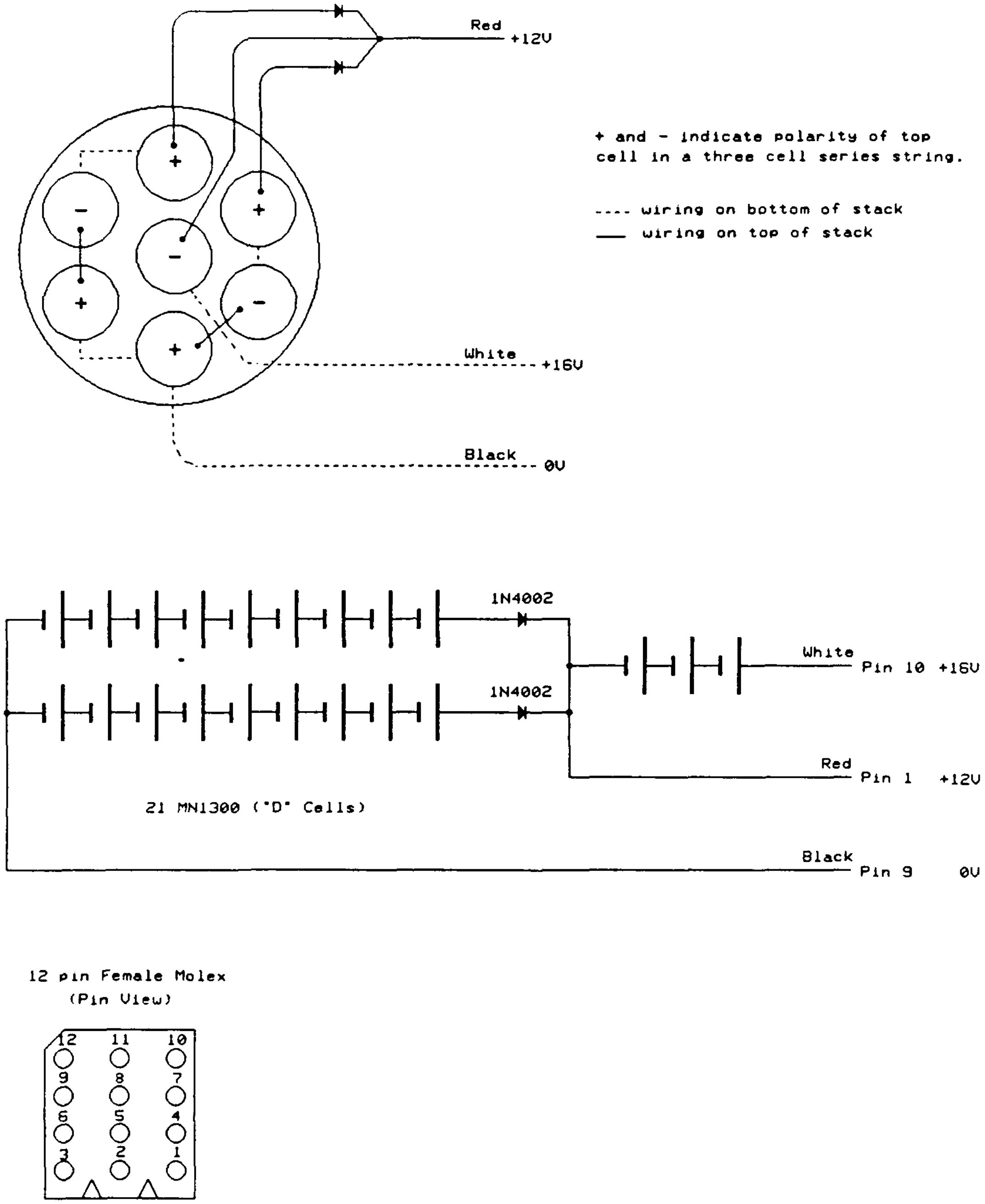
5.2 Back-Plane

Clip the white wire from the pin 5 end of the 10K ohm resistor located on the CAPACITOR card connector between pins 5 and 7. Connect this wire to pin 10 of the LOGIC card connector.

5.3 Battery Stack

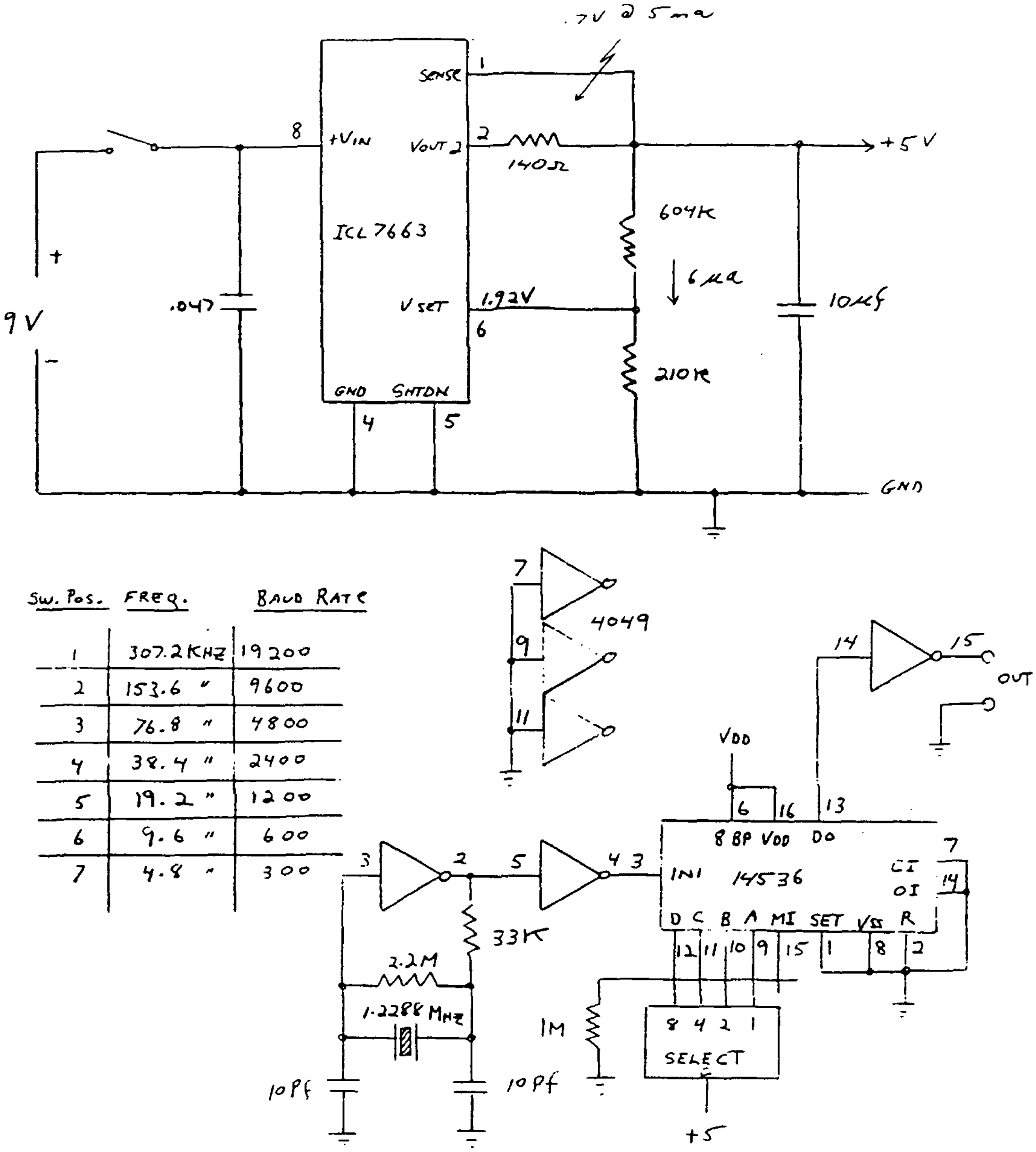
Locate the 12 pin female MOLEX connector which exits the battery housing. Remove the orange wire from pin 1 of this connector, and discard it. Remove the red wire from pin 2 and place it in pin 1. Remove the white/red trace wire from pin 7 and place it in pin 2.

Refer to Figure 7. This is a schematic of the modified stack. Using twenty-one B1300-T2 alkaline cells and two 1N4002 diodes, construct such a stack and connect it to the molex connector as illustrated.



+ and - indicate polarity of top cell in a three cell series string.

Figure 7: Interrogator Battery Pack



Sw. Pos.	Freq.	Baud Rate
1	307.2KHz	19200
2	153.6 "	9600
3	76.8 "	4800
4	38.4 "	2400
5	19.2 "	1200
6	9.6 "	600
7	4.8 "	300

WOODS HOLE OCEANOGRAPHIC INSTITUTION
WOODS HOLE, MASS. 02543

PROJ. 10/379.00 BY LIBERATORE

SHEET 1 OF 1 DATE 6/11/85

TITLE INTERROGATOR
UART CLOCK GENERATOR

SP 11288A

Figure 8 Interrogator UART Clock Generator

6.0 ACKNOWLEDGEMENTS

As a general rule, many hands are involved in the development of an oceanographic instrument and the interrogator was no exception. The author wishes to gratefully acknowledge contributions to this endeavor made by the following people and organizations: Benthos, Inc. of N. Falmouth Mass. for their support during the entire program, Scripps Institution of Oceanography at the University of California for funding the publication of this document, Fred Schuler for his many helpful comments and his aid in de-bugging the prototype, Dick Nowak who developed the measurement synchronization algorithm, Bob Spindel for his encouragement, without which the project would not have been undertaken, and finally John Kemp and Paul Boutin for their assistance during the "wet" tests both at Woods Hole and from the deck of the R/V ERLINE.

Funding was provided by the Office of Naval Research
under contract Numbers N00014-C-82-0152
and N00014-85-C-0379.

7.0 REFERENCES

1. The RCA COS/MOS Integrated Circuits Manual SSD-250B

2. The RCA CMOS-LSI Circuits Manual SSD-260A

3. RCA ICAN-6581 "Power-on Reset/Run circuits for the RCA CDP1802 COSMAC microprocessor"

4. RCA ICAN-6304 "Power Supplies for COS/MOS"

5. RCA ICAN-6525 "Guide to Better Handling and Operation of CMOS Integrated Circuits:

6. RCA ICAN-6576 "Power-Supply Considerations for COS/MOS Devices"

7. The RCA User Manual for the CDP1802 COSMAC Microprocessor MPM-201B

8. Benthos report 0-210-TR17A-GF, "Instructions for the installation, operation, and maintenance of the model 210-TR17A-GF combination commandable transponder and glow flash"

9. Benthos report 0-210-TCSSA, "Instructions for installation, operation and maintenance of the model (ES) 210-TCSSA acoustic transceiver"

10. The Benthos XT-6000 Technical Manual

11. The Motorola CMOS Data Manual

12. The MAXIM Data Acquisition Catalog

8.0 APPENDIX

8.1 Deployment History

During the past five years, the interrogator has been successfully employed to navigate more than twenty moorings set as part of five major Tomography experiments fielded in the North Atlantic, North Pacific, Gulf of Mexico, the Greenland Sea and the Mediterranean.

Twice during the course of these experiments an interrogator has failed. One system recovered from the RTE-88 experiment failed after three months of operation. Interrogator S/N 008 was recovered from the Greenland Sea in 1989 with a completely depleted battery. On inspection a leaky cell in the battery stack was discovered and may have caused the problem. However, both of these failures might also be attributed to a marginal memory component forcing the program to "hang", which in turn would disable power switching and cause the battery to drain at a 6 to 10mA rate. The modification recommended at the end of section 4.4 should help to eliminate this type of failure.

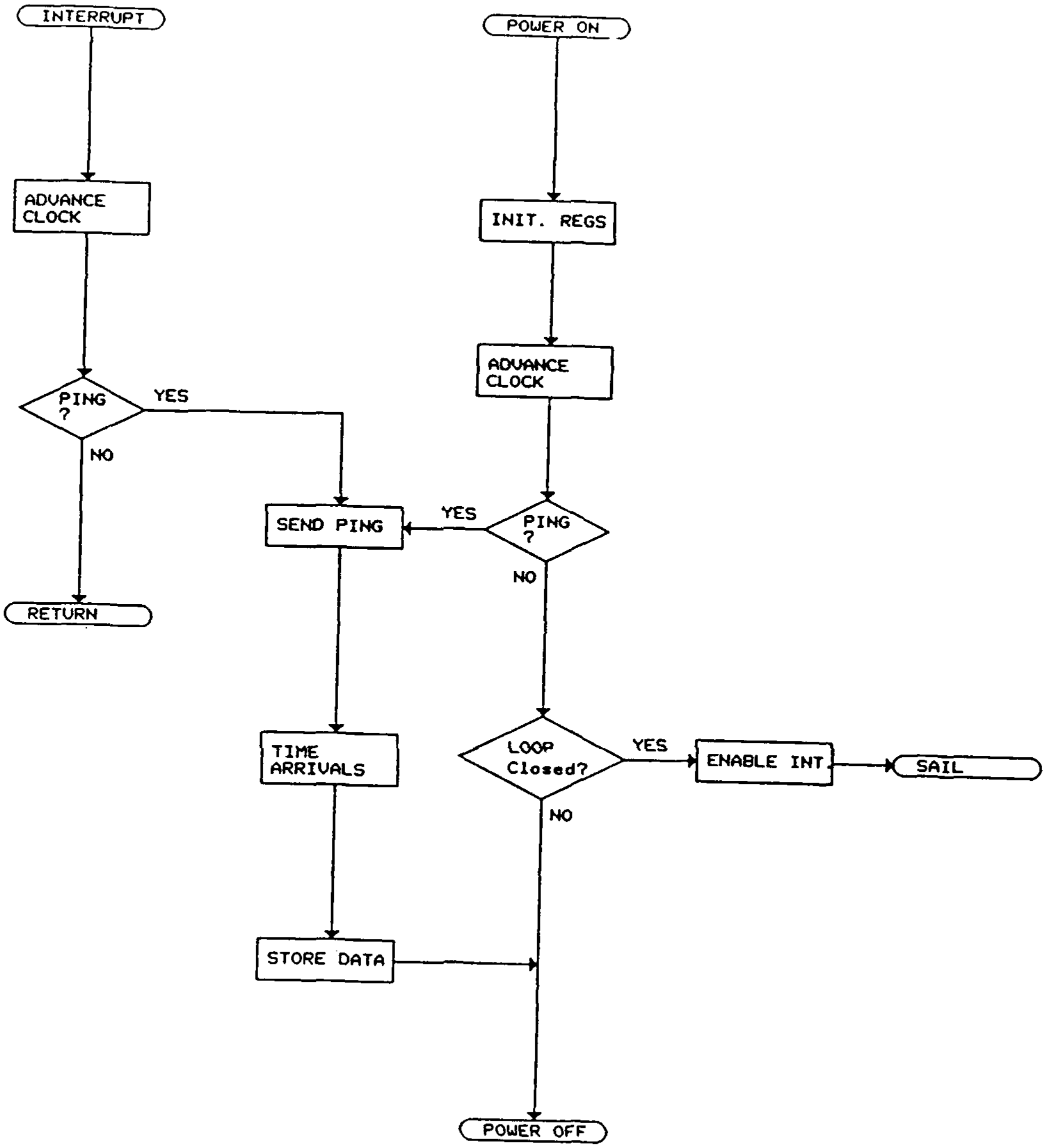


Figure 9 Interrogator Program Flow Chart

8.2 PNAVLGR Program

INTERROGATOR GLOBAL PAGE

LABEL	ADDRESS	FUNCTION	LABEL	ADDRESS	FUNCTION
GLOBAL	FF00	UART STAT. OR CHAR	DSHD	FF30	DEC. STRT.H.D.
	1	SYSTEM ERROR FLAG		1	" " T.D.
	2			2	" " U.D.
	3	USED BY SALTTY		3	" " T.H.
	4	" " "		4	" " U.H.
SCRACH	5			5	" " T.M.
	6	USED BY HTOA	DSUM	6	" " U.M.
	7	" " "		7	--------------
	8	USED BY PHXIN	ASHD	8	ASCII STRT.H.D.
	9	" " "		9	" " T.D.
	A			A	" " U.D.
CRCHI	B	CRC HIGH BYTE		B	" STOP CHAR.
CRCLO	C	CRC LOW BYTE	ASTH	C	" STRT.T.H.
	D			D	" " U.H.
STRADD	E	STORE ADDRESS HI		E	" STOP CHAR.
	F	" " LO	ASTM	F	" STRT T.M.
HD	FF10	DEC. H. DAYS	ASUM	FF40	ASCII STRT.U.M.
	·1	" T. "		1	" STOP CHAR.
	2	" U. "		2	
	3	" T. HOURS	GOFLG	3	GO FLAG HI AA -
	4	" U. "		4	" " LO SET
	5	" T. MINUTES	DIHM	5	DEC. INT. H.M.
	6	" U. "		6	" " T.M.
TICK	7	TICK FLAG		7	" " U.M.
NXTM	8	ASCII H. DAYS		8	
	9	" T. "		9	
	A	" U. "	AIHM	A	ASCII INT.H.M.
	B	ASCII SPACE		B	" " T.M.
	C	" T. HOURS		C	" " U.M.
	D	" U. "		D	" STOP CHAR.
	E	" : CHARACTER		E	
	F	" T. MINUTES		F	
	FF20	ASCII U. MINUTES		FF50	
	1	" STOP CHARACTER		1	
	2			2	
	3			3	
HEXMI	4	HEX MEAS. INT. HI		4	
	5	" " " LO		5	
MINOW	6	HEX MINS.TO NEXT HI		6	
	7	" " " " LO		7	
	8			8	
S1HD	9	DEC. TIME + 1 MIN.		9	
	A	" " T.D.		A	
	B	" " U.D.		B	
	C	" " T.H.		C	
	D	" " U.H.		D	
	E	" " T.M.		E	
S1UM	F	" " U.M.		F	

```
                TITLE       INTERROGATOR CONTROL/DATA LOGGER (PNAVLGR.MAC)
                SUBTTL      WOODS HOLE OCEANOGRAPHIC INST.  OCEAN ENGINEERING
                ;
                ;   Ver. 1.1  27 Feb. 1985            By Steve Liberatore
                ;
                ; Copyright (c) 1985 Woods Hole Oceanographic Institution.
                ; All rights reserved.
                ;
                ;A SAIL compatible micro-power Controller / Data Logger for the
                ;Benthos Quad-M Transceiver. All standard 1802 monitor functions
                ;are implemented along with extensive self test capabilities.
                ;The 60k (F000) RAM memory allows space for 7648 measurements.
                ;The 7649th measurement will not be stored, and will cause the
                ;system to idle. A measurement will consist of a 16 bit time
                ;code word, and three 16 bit two way travel time words. When
                ;output to disk there will be two measurements per line. An LSB
                ;of travel time will be equivalent to 250 uS., and time will be
                ;encoded as follows:
                ;
                ;  BIT #    15 14 13 12 11 10  9  8  7  6  5  4  3  2  1  0
                ; UNITS     HD HD TD TD TD TD UD UD UD UD TH TH UH UH UH UH
                ; WEIGHT     2  1  8  4  2  1  8  4  2  1  2  1  8  4  2  1
                ;
                ;Measurement data will be stored in memory beginning at address
                ;1000H and ordered as follows:
                ;
                ;TIME CODE, TRAVEL TIME A, TRAVEL TIME B, AND TRAVEL TIME C.
                ;
                ;   To assemble this program using the IBM PC/AT system:
                ;
                ;  First execute the Z80MU command to enter the CP/M SHELL
                ;
                ;1)  Type M18 =PNAVLGR.MAC
                ;2)  After the system prompt type L18.
                ;3)  After the LINK prompt type /P:0000
                ;4)  Next type PNAVLGR,PNAVLGR/N/X/E
                ;5)  Answer the MOVE question with N
                ;6)  At the system prompt re-enter DOS by typing E
                ;7)  Use a word processor to divide PNAVLGR.HEX. The
                ;    two new files will be named PROM1.HEX and PROM2.HEX.
                ;    The PROM1 file contains addresses 0 thru 817 and the
                ;    PROM2 file contains addresses 7FC thru FFF.
                ;8)  Next type MOVEHEX.
                ;9)  At the system prompt type UDLINT.
                ;10) To burn the first PROM type PROM1 and to burn the
                ;    second PROM type PROM2.
                ;11) Return to the system by typing BYE
                ;
                ;
        C          INCLUDE IINIT.MAC
        C       ;    *************
        C       ;    * IINIT.MAC *
        C       ;    *************
        C       ;
        C       ;-----------------------------------------------
        C       ;+ THIS SEGMENT OF CODE WILL INITIALIZE ALL REGISTERS +
        C       ;-----------------------------------------------
```

```
                        C       ;
 1000                   C       RAM     EQU     1000H           ;DEFINE START OF RAM
 F000                   C       SIZE    EQU     0F000H          ;DEFINE AMOUNT OF RAM
 FF00                   C       GLOBAL  EQU     RAM+SIZE-0100H  ;DEFINE GLOBAL PAGE
 FFFF                   C       STACK   EQU     RAM+SIZE-01H    ;STACK BEGINS HERE
                        C       ;
 0000'                  C               CSEG                    ;CODE RELATIVE ADDRESS
                        C               ORG     0000H           ;START AT 0000
                        C       ;
 0000'   71             C       INIT:   DIS                     ;DISABLE INTERRUPTS
 0001'   00             C               DB      00H             ;SET X AND P TO 0
 0002'   F8 00          C               LDI     00H             ;LOAD ZEROS
 0004'   A6             C               PLO     R6              ;INTO
 0005'   B6             C               PHI     R6              ;R6 (SCRT LINK)
 0006'   A8             C               PLO     R8              ;AND
 0007'   B8             C               PHI     R8              ;R8
 0008'   A9             C               PLO     R9              ;AND
 0009'   B9             C               PHI     R9              ;R9
 000A'   AB             C               PLO     RB              ;AND
 000B'   BB             C               PHI     RB              ;RB
 000C'   AC             C               PLO     RC              ;AND
 000D'   BC             C               PHI     RC              ;RC
 000E'   AD             C               PLO     RD              ;AND
 000F'   BD             C               PHI     RD              ;RD
 0010'   AE             C               PLO     RE              ;AND
 0011'   BE             C               PHI     RE              ;RE
 0012'   AF             C               PLO     RF              ;AND
 0013'   BF             C               PHI     RF              ;RF
 0014'   F8 03'         C               LDI     HIGH    (INTRPT);SET UP R1 TO BE
 0016'   B1             C               PHI     R1              ;THE INTERRUPT POINTER
 0017'   F8 62'         C               LDI     LOW     (INTRPT)
 0019'   A1             C               PLO     R1
 001A'   F8 FF          C               LDI     HIGH    (STACK) ;SET UP R2 TO BE
 001C'   B2             C               PHI     R2              ;THE STACK POINTER
 001D'   F8 FF          C               LDI     LOW     (STACK)
 001F'   A2             C               PLO     R2
 0020'   F8 03'         C               LDI     HIGH    (CLKTIC);SET UP R3 TO BE THE
 0022'   B3             C               PHI     R3              ;PROGRAM REGISTER
 0023'   F8 A4'         C               LDI     LOW     (CLKTIC)
 0025'   A3             C               PLO     R3
 0026'   F8 00'         C               LDI     HIGH    (CALL)  ;SET UP R4 FOR THE
 0028'   B4             C               PHI     R4              ;CALL ROUTINE POINTER
 0029'   F8 3A'         C               LDI     LOW     (CALL)
 002B'   A4             C               PLO     R4
 002C'   F8 00'         C               LDI     HIGH    (RETURN);SET UP R5 TO BE
 002E'   B5             C               PHI     R5              ;THE RETURN POINTER
 002F'   F8 4B'         C               LDI     LOW     (RETURN)
 0031'   A5             C               PLO     R5
 0032'   F8 FF          C               LDI     HIGH    (GLOBAL);SET UP R7 TO BE
 0034'   B7             C               PHI     R7              ;THE GLOBAL POINTER
 0035'   F8 00          C               LDI     LOW     (GLOBAL)
 0037'   A7             C               PLO     R7
                        C       ;
                        C       ;At this point, all registers are preset, so execution
                        C       ;begins in register R3.
                        C       ;
 0038'   D3                 C            SEP     R3
                            C       ;
                                    ;
                            C            INCLUDE SMACS.MAC
                            C       ;    *************
                            C       ;    * SMACS.MAC *
                            C       ;    *************
                            C       ;
                            C       ;------------------------------------------
                            C       ;+ ALL MACROS CALLED BY SAIL.MAC ARE LISTED HERE +
                            C       ;------------------------------------------
                            C       ;
                            C       ;This MACRO executes the CALL routine
                            C       ;
                            C       CALL     MACRO   SUB              ;BEGIN MACRO CALL
                            C                .SALL                    ;NO LISTING
                            C                SEP     R4               ;CALL
                            C                DW      SUB              ;SUBROUTINE
                            C                ENDM                     ;END MACRO CALL
                            C       ;
                            C       ;This MACRO executes the RETURN routine.
                            C       ;
                            C       RETURN   MACRO                    ;BEGIN MACRO RETURN
                            C                .SALL                    ;NO LISTING
                            C                SEP     R5               ;RETURN
                            C                ENDM                     ;END MACRO RETURN
                            C       ;
                            C       ;This MACRO looks for UART status errors.
                            C       ;
                            C       ERROR?   MACRO                    ;BEGIN MACRO ERROR
                            C                .SALL                    ;NO LISTING
                            C                GHI     RC               ;RECOVER STATUS WORD
                            C                LBNZ    ERVEC            ;BRANCH ON ERROR FLAG
                            C                ENDM
                            C       ;
                            C       ;Here is a MACRO which when called will sequentially
                            C       ;input characters and compare them with a character
                            C       ;string stored in permanent memory. Unsuccessful
                            C       ;comparisons will cause the MACRO to exit with a
                            C       ;non zero result remaining in the ACCUMULATOR.
                            C       ;
                            C       WORD?    MACRO   WORD             ;BEGIN MACRO WORD
                            C                .SALL                    ;NO LISTING
                            C                CALL    COMPAR           ;CALL SUBROUTINE
                            C                DW      WORD             ;PASS WORD
                            C                ERROR?                   ;REACT TO FLAGS
                            C                GLO     RC               ;GET COMPARE RESULT
                            C                ENDM                     ;END OF MACRO WORD?
                            C       ;
                            C       ;Here is a MACRO which when called will input an
                            C       ;ASCII character then exit with that character
                            C       ;remaining in the ACCUMULATOR.
                            C       ;
                            C       CHAR?    MACRO                    ;BEGIN MACRO CHAR?
                            C                .SALL                    ;NO LISTING
                            C                CALL    INCHAR           ;CALL SUBROUTINE INCHAR
```

```
                        C               ERROR?                  ;REACT TO FLAGS
                        C               GLO     RC              ;RECOVER CHARACTER
                        C               ENDM                    ;END MACRO DA?
                        C       ;
                        C       ;Here is a MACRO which will call SALTTY, pass the
                        C       ;message address, and react to errors upon exiting.
                        C       ;
                        C       TYPMSG  MACRO   MSG             ;BEGIN MACRO TYPMSG
                        C               .SALL                   ;NO LISTING
                        C               CALL    SALTTY          ;CALL SUBROUTINE SALTTY
                        C               DW      MSG             ;PASS MESSAGE
                        C               ERROR?                  ;REACT TO FLAGS
                        C               ENDM                    ;END MACRO TYPMSG
                        C       ;
                        C       ;This MACRO recovers the SYSTEM FLAG. This flag is
                        C       ;stored in RAM one location higher than the character
                        C       ;flag. Bit 0 indicates whether or not the system is
                        C       ;LOCKED, and bit 7 is used by the CRC routine.
                        C       ;The remaining bits may be user defined.
                        C       ;
                        C       GETFLG  MACRO                   ;BEGIN MACRO GETFLG
                        C               .SALL                   ;NO LISTING
                        C               LDI     01H             ;POINT TO FLAG
                        C               PLO     R7
                        C               LDN     R7              ;GET FLAG
                        C               ENDM                    ;END MACRO GETFLG
                        C       ;
                                ;
                        C               INCLUDE SCRT.MAC
                        C       ;       ************
                        C       ;       * SCRT.MAC *
                        C       ;       ************
                        C       ;
                        C       ;--------------------------------------------------------
                        C       ;+THESE ARE THE RCA STANDARD CALL AND RETURN ROUTINES.+
                        C       ;--------------------------------------------------------
                        C       ;
                        C       ;
                        C       ;THIS IS THE CALL ROUTINE, IT RUNS IN R4
                        C       ;
0039'   D3              C       EXITC:  SEP     R3      ;R3 IS POINTING AT THE FIRST
                        C                               ;INSTRUCTION IN THE SUBROUTINE
003A'                   C       CALL::                  ;THIS IS A "PUBLIC ROUTINE"
003A'   E2              C               SEX     R2      ;POINT TO THE STACK
003B'   96              C               GHI     R6      ;PUSH R6 ON TO THE STACK
003C'   73              C               STXD            ;AND PREPARE IT TO POINT TO
003D'   86              C               GLO     R6      ;ARGUMENTS. THEN DECREMENT
003E'   73              C               STXD            ;TO A FREE LOCATION
003F'   93              C               GHI     R3      ;COPY R3 TO R6
0040'   B6              C               PHI     R6
0041'   83              C               GLO     R3
0042'   A6              C               PLO     R6
0043'   46              C               LDA     R6      ;GET THE SUBROUTINE ADDRESS
0044'   B3              C               PHI     R3      ;AND PASS IT TO R3
0045'   46              C               LDA     R6
0046'   A3              C               PLO     R3
```

```
0047'   C0 0039'    C            LBR     EXITC   ;RUN THE SUBROUTINE IN R3
                    C     ;
                    C     ;THIS IS THE RETURN ROUTINE, IT RUNS IN R5
                    C     ;
004A'   D3          C     EXITR: SEP     R3      ;RETURN TO MAIN PROGRAM
004B'               C     RETURN::               ;THIS IS A "PUBLIC ROUTINE"
004B'   96          C            GHI     R6      ;COPY R6 INTO R3
004C'   B3          C            PHI     R3      ;R3 CONTAINS THE RETURN
004D'   86          C            GLO     R6      ;ADDRESS
004E'   A3          C            PLO     R3
004F'   E2          C            SEX     R2      ;POINT TO THE STACK
0050'   12          C            INC     R2      ;GET OLD VALUE OF R6
0051'   72          C            LDXA            ;AND RESTORE IT TO R6
0052'   A6          C            PLO     R6
0053'   F0          C            LDX
0054'   B6          C            PHI     R6
0055'   C0 004A'    C            LBR     EXITR   ;RUN MAIN PROGRAM
                    C     ;
                          ;
                    C            INCLUDE ATOH.MAC
                    C     ;      ************
                    C     ;      * ATOH.MAC *
                    C     ;      ************
                    C     ;
                    C     ;      ------------------------------------
                    C     ;      * ASCII TO HEXADECIMAL CONVERTER *
                    C     ;      ------------------------------------
                    C     ;              (RC)
                    C     ;
                    C     ;This sub-routine converts the ASCII character in the
                    C     ;low half of RC to a HEX digit, shifts this hex digit
                    C     ;four (4) places to the left and returns with it in
                    C     ;the low half of RC.
                    C     ;
0058'   8C          C     ATOH:: GLO     RC              ;GET THE ASCII CHAR.
0059'   FF 30       C            SMI     "0"             ;TOO SMALL ?
005B'   CB 0081'    C            LBNF    AERROR          ;IF SO GOTO ERROR
005E'   BC          C            PHI     RC              ;SAVE RESULT
005F'   8C          C            GLO     RC              ;RESTORE
0060'   FF 47       C            SMI     "G"             ;TOO LARGE ?
0062'   C3 0081'    C            LBDF    AERROR          ;IF SO GOTO ERROR
0065'   9C          C            GHI     RC              ;CHAR MINUS ASCII BIAS
0066'   FF 0A       C            SMI     0AH             ;IS IT 0 THROUGH 9 ?
0068'   CB 0077'    C            LBNF    HDONE           ;IF SO CONVERT IS DONE
006B'   9C          C            GHI     RC              ;RESTORE
006C'   FF 11       C            SMI     11H             ;IS IT ASCII ?
006E'   CB 0081'    C            LBNF    AERROR          ;IF NOT GOTO ERROR
0071'   9C          C            GHI     RC              ;RESTORE
0072'   FF 07       C            SMI     07H             ;REMOVE ALPHA BIAS
0074'   C0 0078'    C            LBR     SHIFT           ;GOTO SHIFT
0077'   9C          C     HDONE: GHI     RC              ;RESTORE
0078'   FE          C     SHIFT: SHL                     ;SHIFT LEFT 4 TIMES
0079'   FE          C            SHL
007A'   FE          C            SHL
007B'   FE          C            SHL
007C'   AC          C            PLO     RC              ;HEX DIGIT TO RC LOW
```

```
 007D'   F8 00          C              LDI     00H             ;CLEAR ERROR FLAGS
 007F'   BC             C              PHI     RC              ;AND NON-HEX FLAG
                        C              RETURN                  ;BACK TO MAIN
 0080'   D5             C+
 0081'   F8 01          C     AERROR:  LDI     01H             ;SET NON-HEX FLAG
 0083'   BC             C              PHI     RC
                        C              RETURN                  ;BACK TO MAIN
 0084'   D5             C+
                        C     ;
                              ;
                        C              INCLUDE HTOA.MAC
                        C     ;        ************
                        C     ;        * HTOA.MAC *
                        C     ;        ************
                        C     ;
                        C     ;        ______________________________
                        C     ;        + HEXADECIMAL TO ASCII CONVERTER +
                        C     ;        ______________________________
                        C     ;                (RC)
                        C     ;
                        C     ;This sub-routine converts the HEX digit in the
                        C     ;low half of RC to an ASCII character, and returns
                        C     ;with this character in the high half of RC.
                        C     ;
 0085'   8C             C     HTOA::   GLO     RC              ;GET THE HEX DIGIT
 0086'   FA 0F          C              ANI     0FH             ;MASK HIGH BYTE
 0088'   FC 30          C              ADI     30H             ;ADD ASCII BIAS
 008A'   BC             C              PHI     RC              ;SAVE RESULT
 008B'   FF 3A          C              SMI     3AH             ;IS IT NUMERIC ?
 008D'   CB 0094'       C              LBNF    ADONE           ;IF SO CONVERT IS DONE
 0090'   9C             C              GHI     RC              ;OTHERWISE,
 0091'   FC 07          C              ADI     07H             ;ADD ALPHA BIAS
 0093'   BC             C              PHI     RC              ;SAVE RESULT
 0094'                  C     ADONE:   RETURN                  ;RETURN TO MAIN
 0094'   D5             C+
                        C     ;
                              ;
                        C              INCLUDE DTOA.MAC
                        C     ;        ************
                        C     ;        * DTOA.MAC *
                        C     ;        ************
                        C     ;
                        C     ;______________________
                        C     ;+ ADD ASCII BIAS AND STORE +
                        C     ;______________________
                        C     ;                (RC)
                        C     ;This subroutine will add 30 hex to the byte
                        C     ;pointed at by R7, store the result using RA
                        C     ;as a pointer, then increment the pointers.
                        C     ;This operation will be repeated n times as
                        C     ;specified by the in-line byte following the
                        C     ;call instruction.
                        C     ;
 0095'   46             C     DTOA:    LDA     R6              ;GET REPEAT VALUE
 0096'   AC             C              PLO     RC              ;SET COUNTER
 0097'   47             C     ADBIAS:  LDA     R7              ;GET DIGIT
```

```
0098'   FC 30       C            ADI     30H              ;ADD ASCII DEC. BIAS
009A'   5A          C            STR     RA               ;STORE RESULT
009B'   1A          C            INC     RA               ;ADVANCE POINTER
009C'   2C          C            DEC     RC               ;COUNT OPERATION
009D'   8C          C            GLO     RC               ;TEST COUNTER
009E'   CA 0097'    C            LBNZ    ADBIAS           ;EXIT IF DONE
                    C            RETURN                   ;OTHERWISE CONTINUE
00A1'   D5          C+
                    C     ;
                          ;
                    C            INCLUDE DELAY.MAC
                    C     ;              *************
                    C     ;              * DELAY.MAC *
                    C     ;              *************
                    C     ;                  (RC)
                    C     ; -----------------------------
                    C     ; + DELAY PROGRAM EXECUTION +
                    C     ; -----------------------------
                    C     ;
                    C     ;This subroutine will delay program execution by an amount
                    C     ;of time equivalent to (120N +168)/Fc where Fc is the system
                    C     ;clock frequency and N is a two byte value specified by the
                    C     ;bytes following the call instruction. SCRT is expected.
                    C     ;
00A2'   46          C     DELAY: LDA     R6               ;LOAD DELAY CONSTANT
00A3'   BC          C            PHI     RC               ;USE RC AS A DOWN COUNTER
00A4'   46          C            LDA     R6
00A5'   AC          C            PLO     RC
00A6'   9C          C     TSTHIC: GHI    RC               ;CONTINUE TESTING RC
00A7'   CA 00B2'    C            LBNZ    WST5             ;WHEN EQUAL TO ZERO
00AA'   8C          C            GLO     RC               ;SPECIFIED TIME HAS
00AB'   C2 00B6'    C            LBZ     EXDLY            ;ELAPSED, OTHERWISE
00AE'   2C          C     DECC:  DEC     RC               ;DEC COUNTER AND TEST FOR
00AF'   C0 00A6'    C            LBR     TSTHIC           ;NON ZERO VALUE
00B2'   8C          C     WST5:  GLO     RC               ;EQUALIZE COUNTER LOOPS
00B3'   C0 00AE'    C            LBR     DECC             ;AND CONTINUE
00B6'               C     EXDLY: RETURN                   ;RETURN TO MAIN WHEN COUNT = 0000
00B6'   D5          C+
                    C     ;
                    C
                          ;
                    C            INCLUDE RSB2A.MAC
                    C     ;              *************
                    C     ;              * RSB2A.MAC *
                    C     ;              *************
                    C     ;
                    C     ;------------------------------------
                    C     ;+ RIGHT SHIFT n BITS FROM RB TO RA +
                    C     ;------------------------------------
                    C     ;              (RA, RB, RC)
                    C     ;
                    C     ;This subroutine will right shift a single bit
                    C     ;from register B to register A. This operation
                    C     ;will be repeated a number of times as specified
                    C     ;by the byte following the call instruction.
                    C     ;
```

```
00B7'   46          C     RSB2A:  LDA     R6              ;GET REPEAT VALUE
00B8'   AC          C             PLO     RC              ;USE RC AS A COUNTER
00B9'   9B          C     SHRB:   GHI     RB              ;RIGHT SHIFT A BIT
00BA'   F6          C             SHR                     ;FROM RB TO RA
00BB'   BB          C             PHI     RB
00BC'   8B          C             GLO     RB
00BD'   76          C             RSHR
00BE'   AB          C             PLO     RB
00BF'   9A          C             GHI     RA
00C0'   76          C             RSHR
00C1'   BA          C             PHI     RA
00C2'   8A          C             GLO     RA
00C3'   76          C             RSHR
00C4'   AA          C             PLO     RA
00C5'   2C          C             DEC     RC              ;DECREMENT COUNTER
00C6'   8C          C             GLO     RC              ;TEST FOR ZERO AND
00C7'   CA 00B9'    C             LBNZ    SHRB            ;IF FOUND
                    C             RETURN                  ;RETURN TO MAIN
00CA'   D5          C+
                    C     ;
                          ;
                    C             INCLUDE SALTTY.MAC
                    C     ;       **************
                    C     ;       * SALTTY.MAC *
                    C     ;       **************
                    C     ;
                    C     ; ________________________________
                    C     ; + TYPES MESSAGE NAMED AFTER CALL +
                    C     ; ________________________________
                    C     ;               (RC)
                    C     ;
                    C     ;This subroutine will type the message indicated
                    C     ;by the call. The loop will be continually monitored,
                    C     ;Any error condition will cause this routine to
                    C     ;exit. The status word and the last character typed
                    C     ;will be available in RC upon exiting this routine.
                    C     ;
00CB'               C     SALTTY::                        ;A PUBLIC ROUTINE
00CB'   E7          C             SEX     R7              ;USE R7 AS THE POINTER
00CC'   F8 04       C             LDI     LOW (SCRACH-1)  ;POINT TO A SCRATCH
00CE'   A7          C             PLO     R7              ;LOCATION IN RAM
00CF'   8A          C             GLO     RA              ;SAVE OLD ADDRESS
00D0'   73          C             STXD                    ;POINTER
00D1'   9A          C             GHI     RA
00D2'   73          C             STXD
00D3'   46          C             LDA     R6              ;GET HIGH HALF OF
00D4'   BA          C             PHI     RA              ;MESSAGE ADDRESS
00D5'   46          C             LDA     R6              ;GET LOW HALF OF
00D6'   AA          C             PLO     RA              ;MESSAGE ADDRESS
                    C     ;
                    C     ;Enter the subroutine here if the address of the data
                    C     ;to type is already in register RA.
                    C     ;
00D7'   E7          C     ITYPE:: SEX     R7              ;USE R7 AS THE POINTER
00D8'   F8 00       C             LDI     00H             ;POINT TO A SCRATCH
00DA'   A7          C             PLO     R7              ;LOCATION IN RAM
```

```
 00DB'    6E          C            INP    DATA             ;CLEAR UART DA BIT
 00DC'    E7          C     THRE?: SEX    R7               ;RESET POINTER TO R7
 00DD'    6F          C            INP    STATUS           ;GET UART STATUS
 00DE'    FA 10       C            ANI    10H              ;IS THE LOOP OPEN ?
 00E0'    C2 00E9'    C            LBZ    TSTHR            ;IF NOT TEST FOR THRE
 00E3'    F8 80       C            LDI    80H              ;OTHERWISE,
 00E5'    BC          C            PHI    RC               ;SET FLAG
 00E6'    C0 010B'    C            LBR    TXIT             ;AND RETURN
 00E9'    6F          C     TSTHR: INP    STATUS           ;GET UART STATUS
 00EA'    FE          C            SHL                     ;IS THE THRE ?
 00EB'    CB 00DC'    C            LBNF   THRE?            ;IF NOT, KEEP TRYING
 00EE'    0A          C            LDN    RA               ;GET NEXT CHARACTER
 00EF'    FB 7E       C            XRI    STOP             ;MESSAGE OVER ?
 00F1'    C2 010B'    C            LBZ    TXIT             ;IF SO EXIT
 00F4'    EA          C            SEX    RA               ;OTHERWISE, TYPE THE
 00F5'    66          C            OUT    DATA             ;CHARACTER
                      C            CALL   INCHAR           ;MONITOR THE LOOP FOR
 00F6'    D4          C+
 00F7'    0145'       C+
 00F9'    9C          C            GHI    RC               ;ANY ERRORS ?
 00FA'    CA 010B'    C            LBNZ   TXIT             ;IF SO, EXIT
 00FD'    2A          C            DEC    RA               ;WAS THE LAST CHAR.
 00FE'    EA          C            SEX    RA               ;TYPED THE SAME AS
 00FF'    8C          C            GLO    RC               ;THE CHARACTER JUST
 0100'    F3          C            XOR                     ;RECEIVED ?
 0101'    CA 0108'    C            LBNZ   BADCHR           ;IF SO,
 0104'    1A          C            INC    RA               ;REPOSITION POINTER
 0105'    C0 00DC'    C            LBR    THRE?            ;AND CONTINUE
 0108'    F8 02       C     BADCHR: LDI   02H              ;OTHERWISE, SET FLAG
 010A'    BC          C            PHI    RC
 010B'    F8 C3       C     TXIT:  LDI    LOW (SCRACH-2)   ;RESTORE OLD ADDRESS
 010D'    A7          C            PLO    R7               ;POINTER
 010E'    47          C            LDA    R7
 010F'    BA          C            PHI    RA
 0110'    47          C            LDA    R7
 0111'    AA          C            PLO    RA
                      C            RETURN                  ;AND RETURN
 0112'    D5          C+
                      C     ;
                            ;
                      C            INCLUDE ASKOK.MAC
                      C     ;             *************
                      C     ;             * ASKOK.MAC *
                      C     ;             *************
                      C     ;
                      C     ;
                      C     ; ---------------------------------------------
                      C     ; + ASK FOR FINAL PERMISSION TO CARRY OUT A COMMAND +
                      C     ; ---------------------------------------------
                      C     ;
                      C     ;Type OK ? (Y/N) and input a response. Set RC.0 to 00
                      C     ;upon detecting a "Y". Exit upon detecting any error
                      C     ;with the UART status word remaining in RC.1.
                      C     ;
 0113'                C     ASKOK: CALL   SALTTY           ;ASK OK ?
 0113'    D4          C+
 0114'    00CB'       C+
 0116'  0423'        C             DW      OK?
 0118'  9C           C             GHI     RC             ;LOOK FOR UART ERRORS
 0119'  CA 012A'     C             LBNZ    EXASK          ;EXIT IF FOUND
                     C             CALL    INCHAR         ;GET RESPONSE
 011C'  D4           C+
 011D'  0145'        C+
 011F'  9C           C             GHI     RC             ;LOOK FOR UART ERRORS
 0120'  CA 012A'     C             LBNZ    EXASK          ;EXIT IF FOUND
 0123'  8C           C             GLO     RC             ;WAS THE RESPONSE A "Y" ?
 0124'  FB 59        C             XRI     "Y"            ;IF NOT EXIT
 0126'  CA 012A'     C             LBNZ    EXASK          ;OTHERWISE, SET RC.0
 0129'  AC           C             PLO     RC             ;TO 00 AND EXIT
 012A'               C     EXASK:  RETURN                 ;TO SAIL
 012A'  D5           C+
                           ;
                     C             INCLUDE COMPAR.MAC
                     C     ;       ***************
                     C     ;       * COMPAR.MAC *
                     C     ;       ***************
                     C     ;
                     C     ;------------------------------------------
                     C     ;+ COMPARE RECEIVED STRING WITH STORED STRING +
                     C     ;------------------------------------------
                     C     ;                 (RA + RC)
                     C     ;
                     C     ;This subroutine will sequentially input characters
                     C     ;and compare them with a character string stored in
                     C     ;permanent memory. Unsuccessful comparisons will
                     C     ;cause the subroutine to exit leaving a non-zero
                     C     ;result in the low half of register C
                     C     ;
 012B'  46           C     COMPAR: LDA     R6             ;GET ADDRESS OF
 012C'  BA           C             PHI     RA             ;WORD TO COMPARE
 012D'  46           C             LDA     R6             ;USE RA AS
 012E'  AA           C             PLO     RA             ;CHARACTER POINTER
 012F'               C     CTST:   CALL    INCHAR         ;GET A CHARACTER
 012F'  D4           C+
 0130'  0145'        C+
 0132'  9C           C             GHI     RC             ;LOOK FOR FLAGS
 0133'  CA 0144'     C             LBNZ    CMPXIT         ;IF FOUND EXIT
 0136'  8C           C             GLO     RC             ;OTHERWISE, COMPARE
 0137'  EA           C             SEX     RA             ;WITH STORED CHARACTER
 0138'  F3           C             XOR                    ;ARE THEY THE SAME ?
 0139'  CA 0143'     C             LBNZ    DIFFER         ;IF NOT EXIT
 013C'  1A           C             INC     RA             ;WAS THAT THE LAST
 013D'  F8 7E        C             LDI     STOP           ;CHARACTER TO BE
 013F'  F3           C             XOR                    ;COMPARED ?
 0140'  CA 012F'     C             LBNZ    CTST           ;IF NOT CONTINUE
 0143'  AC           C     DIFFER: PLO     RC             ;ELSE SET COMPARE
 0144'               C     CMPXIT: RETURN                 ;FLAG AND RETURN
 0144'  D5           C+
                     C     ;
                           ;
                     C             INCLUDE INCHAR.MAC
                     C     ;       ***************
                     C     ;       * INCHAR.MAC *
```

```
                        C       ;       ***************
                        C       ;
                        C       ; ----------------------------------------------
                        C       ;+INPUT CHARACTER AND UART STATUS FLAG TO REGISTER C +
                        C       ; ----------------------------------------------
                        C       ;               (RC)
                        C       ;
                        C       ;This subroutine will monitor the UART until
                        C       ;either the LOOP is open or data is available.
                        C       ;If data is available, the character will be
                        C       ;input and placed in the low half of RC. The
                        C       ;high half of RC will contain the status flag.
                        C       ;
0145'                   C       INCHAR::                        ;A PUBLIC ROUTINE
0145'   F8 00           C               LDI     00              ;POINT TO A SCRATCH
0147'   A7              C               PLO     R7              ;LOCATION
0148'   BC              C               PHI     RC              ;RESET STATUS FLAGS
0149'   E7              C               SEX     R7              ;X POINTS TO SCRATCH
014A'   6F              C       NDA:    INP     STATUS          ;GET UART STATUS
014B'   FA 10           C               ANI     10H             ;IS THE LOOP CLOSED ?
014D'   C2 0154'        C               LBZ     TSTDA           ;IF SO TEST FOR DA
0150'   F8 80           C               LDI     80H             ;OTHERWISE,
0152'   BC              C               PHI     RC              ;SET ERROR FLAG
                        C               RETURN                  ;AND RETURN
0153'   D5              C+
0154'   07              C       TSTDA:  LDN     R7              ;RESTORE STATUS
0155'   F6              C               SHR                     ;IS DATA AVAILABLE ?
0156'   CB 014A'        C               LBNF    NDA             ;IF NOT, TRY AGAIN
0159'   FA 05           C               ANI     05H             ;OTHERWISE,LOOK FOR
015B'   C2 0162'        C               LBZ     DATAIN          ;ERRORS
015E'   F8 20           C               LDI     20H             ;IF FOUND SET FLAG
0160'   BC              C               PHI     RC              ;AND
                        C               RETURN                  ;EXIT. OTHERWISE,
0161'   D5              C+
0162'   6E              C       DATAIN: INP     DATA            ;GET THE CHARACTER
0163'   AC              C               PLO     RC              ;AND PLACE IN RC LOW
0164'   FB 23           C               XRI     "#"             ;IS IT A "#"
0166'   CA 016C'        C               LBNZ    DADONE          ;IF NOT RETURN
0169'   F8 40           C               LDI     40H             ;OTHERWISE SET
016B'   BC              C               PHI     RC              ;FLAG THEN
016C'                   C       DADONE: RETURN                  ;RETURN TO MAIN
016C'   D5              C+
                        C       ;
                                ;
                        C               INCLUDE GETHEX.MAC
                        C       ;       ***************
                        C       ;       * GETHEX.MAC *
                        C       ;       ***************
                        C       ;
                        C       ; ------------------------------------
                        C       ; + LOAD RB WITH A FOUR DIGIT HEX NUMBER +
                        C       ; ------------------------------------
                        C       ;               (RB + RC)
                        C       ;
                        C       ;This subroutine will load RB with a four digit hex
                        C       ;number. Only the last four hex digits typed are
```

```
                      C     ;entered. Non hex entries will cause this routine
                      C     ;to exit.
                      C     ;
016D'                 C     GETHEX::                    ;THIS WILL BE A PUBLIC
016D'  F8 00          C             LDI     00H         ;CLEAR REGISTER B
016F'  AB             C             PLO     RB
0170'  BB             C             PHI     RB
0171'                 C     GETCHR: CALL    INCHAR      ;GET A CHARACTER
0171'  D4             C+
0172'  0145'          C+
0174'  9C             C             GHI     RC          ;TEST UART FOR ERRORS
0175'  CA 0195'       C             LBNZ    XGETH       ;IF FOUND EXIT
                      C             CALL    ATOH        ;CONVERT ASCII TO HEX
0178'  D4             C+
0179'  0058'          C+
017B'  9C             C             GHI     RC          ;LOOK FOR NON-HEX ENTRY
017C'  CA 0195'       C             LBNZ    XGETH       ;IF FOUND EXIT, ELSE
017F'  8C             C             GLO     RC          ;TRANSFER HEX DIGIT
0180'  BC             C             PHI     RC          ;TO HIGH HALF OF RC
0181'  F8 04          C             LDI     04          ;PREPARE TO SHIFT
0183'  AC             C             PLO     RC          ;HEX CHARACTER TO RB
0184'  9C             C     SHIFTC: GHI     RC          ;BEGIN SHIFT
0185'  FE             C             SHL
0186'  BC             C             PHI     RC
0187'  8B             C             GLO     RB
0188'  7E             C             RSHL
0189'  AB             C             PLO     RB
018A'  9B             C             GHI     RB
018B'  7E             C             RSHL
018C'  BB             C             PHI     RB
018D'  2C             C             DEC     RC          ;IS THIS THE FOURTH
018E'  8C             C             GLO     RC          ;SHIFT ?
018F'  CA 0184'       C             LBNZ    SHIFTC      ;IF NOT SHIFT AGAIN
0192'  C0 0171'       C             LBR     GETCHR      ;ELSE GET NEXT DIGIT
0195'                 C     XGETH:  RETURN              ;EXIT
0195'  D5             C+
                      C     ;
                            ;
                      C             INCLUDE INDEC.MAC
                      C     ;               **************
                      C     ;               * INDEC.MAC *
                      C     ;               **************
                      C     ;
                      C     ;
                      C     ;         + INPUT AND STORE DECIMAL NUMBERS +
                      C     ;
                      C     ;                    (R9,RA,RB,RC,RD)
                      C     ;
                      C     ;This subroutine will input and store n decimal digits
                      C     ;beginning at the address specified by the two  bytes
                      C     ;following the call instruction. The number of bytes
                      C     ;to store is specified by the single byte following
                      C     ;the call. Only the last n digits typed will be stored.
                      C     ;Errors and non-decimal entries cause an exit which
                      C     ;leaves the status word in RC.1 and the last digit
                      C     ;type in RC.0. NOTE: n may not be greater than 4.
```

```
                        C
0196'   46              C       INDEC:  LDA     R6              ;GET STORE ADDRESS
0197'   BD              C               PHI     RD              ;AND PLACE IN RD
0198'   46              C               LDA     R6
0199'   AD              C               PLO     RD
019A'   F8 00           C               LDI     00H             ;ZERO REGISTER B
019C'   AB              C               PLO     RB
019D'   9B              C               PHI     RB
019E'                   C       GETDEC: CALL    INCHAR          ;GET A CHARACTER
019E'   D4              C+
019F'   0145'           C+
01A1'   9C              C               GHI     RC              ;TEST FOR ERRORS
01A2'   CA 01C9'        C               LBNZ    XINDEC          ;EXIT IF ERROR IS FOUND
                        C               CALL    ATOH            ;CONVERT TO HEX
01A5'   D4              C+
01A6'   0058'           C+
01A8'   9C              C               GHI     RC              ;EXIT IF NOT HEX
01A9'   CA 01C9'        C               LBNZ    XINDEC
01AC'   8C              C               GLO     RC              ;TEST FOR DECIMAL
01AD'   FF A0           C               SMI     0A0H            ;AND SET ERROR FLAG
01AF'   C3 01C6'        C               LBDF    XINE            ;IF NOT DECIMAL
01B2'   F8 04           C               LDI     04              ;OTHERWISE, USING R9
01B4'   A9              C               PLO     R9              ;AS A COUNTER, SHIFT
01B5'   8C              C       SHFTC:  GLO     RC              ;THE DIGIT A BIT AT
01B6'   FE              C               SHL                     ;TIME TO RB.
01B7'   AC              C               PLO     RC
01B8'   8B              C               GLO     RB
01B9'   7E              C               RSHL
01BA'   AB              C               PLO     RB
01BB'   9B              C               GHI     RB
01BC'   7E              C               RSHL
01BD'   BB              C               PHI     RB
01BE'   29              C               DEC     R9
01BF'   89              C               GLO     R9              ;TEST FOR FOURTH SHIFT
01C0'   CA 01B5'        C               LBNZ    SHFTC           ;SHIFT AGAIN IF NOT DONE
01C3'   C0 019E'        C               LBR     GETDEC          ;OTHERWISE, GET NEXT DIGIT
                        C       ;
01C6'   F8 01           C       XINE:   LDI     01H             ;INDICATE NON-DECIMAL
01C8'   BC              C               PHI     RC              ;AND RETURN TO SAIL
01C9'   22              C       XINDEC: DEC     R2              ;SAVE THE CONTENTS
01CA'   8C              C               GLO     RC              ;OF REGISTER C
01CB'   73              C               STXD
01CC'   9C              C               GHI     RC
01CD'   73              C               STXD
01CE'   46              C               LDA     R6              ;GET NUMBER OF DIGITS
01CF'   A9              C               PLO     R9              ;TO STORE
01D0'                   C       STRDEC: CALL    RSB2A           ;SHIFT A DIGIT TO RA
01D0'   D4              C+
01D1'   00B7'           C+
01D3'   04              C               DB      04H
01D4'   9A              C               GHI     RA              ;SHIFT TO LSB
01D5'   F6              C               SHR
01D6'   F6              C               SHR
01D7'   F6              C               SHR
01D8'   F6              C               SHR
01D9'   5D              C               STR     RD              ;STORE DECIMAL DIGIT
```

```
01DA'   2D          C           DEC     RD                  ;ADJUST POINTER
01DB'   29          C           DEC     R9
01DC'   89          C           GLO     R9                  ;STORED N DIGITS ?
01DD'   CA 01D0'    C           LBNZ    STRDEC              ;IF SO RESTORE RC
01E0'   12          C           INC     R2
01E1'   42          C           LDA     R2
01E2'   BC          C           PHI     RC
01E3'   42          C           LDA     R2
01E4'   AC          C           PLO     RC
                                RETURN                      ;RETURN TO SAIL;
01E5'   D5           +
                    C           INCLUDE PHXIN.MAC
                    C     ;     *************
                    C     ;     * PHXIN.MAC *
                    C     ;     *************
                    C     ;
                    C     ; ________________________________________
                    C     ; + PROMPTS FOR AND LOADS RB WITH A HEX NUMBER +
                    C     ; ________________________________________
                    C     ;                (RC)
                    C     ;This subroutine will prompt the operator for
                    C     ;a hex number using the prompt message addressed
                    C     ;by the in-line code after the call instruction.
                    C     ;Only the last four hex digits typed are entered.
                    C     ;UART errors and non-hex entries cause this routine
                    C     ;to exit with a non-zero value remaining in RC.1
                    C     ;
01E6'               C     PHXIN::                                     ;A PUBLIC ROUTINE
01E6'   E7          C           SEX     R7                            ;USING R7 AS A POINTER
01E7'   F8 09       C           LDI     LOW     (SCRACH+4)            ;SAVE THE CONTENTS OF
01E9'   A7          C           PLO     R7                            ;REGISTER RA
01EA'   8A          C           GLO     RA                            ;FIRST THE LOW HALF
01EB'   73          C           STXD
01EC'   9A          C           GHI     RA
01ED'   73          C           STXD                                  ;THEN THE HIGH HALF
01EE'   46          C           LDA     R6                            ;GET HIGH HALF OF
01EF'   BA          C           PHI     RA                            ;MESSAGE ADDRESS
01F0'   46          C           LDA     R6                            ;GET LOW HALF OF
01F1'   AA          C           PLO     RA                            ;MESSAGE ADDRESS
                    C           CALL    ITYPE                         ;TYPE THE MESSAGE
01F2'   D4          C+
01F3'   00D7'       C+
01F5'   9C          C           GHI     RC                            ;LOOK FOR UART ERRORS
01F6'   CA 0211'    C           LBNZ    EXPHXN                        ;EXIT IF FOUND
                    C           CALL    GETHEX                        ;GET THE HEX NUMBER
01F9'   D4          C+
01FA'   016D'       C+
01FC'   9C          C           GHI     RC                            ;GET THE STATUS WORD
01FD'   FA FE       C           ANI     0FEH                          ;MASK NON-HEX FLAG
01FF'   CA 0211'    C           LBNZ    EXPHXN                        ;EXIT ON UART ERRORS
0202'   8C          C           GLO     RC                            ;WAS THE NON-HEX ENTRY
0203'   FB 20       C           XRI     SPACE                         ;A SPACE ?
0205'   C2 0219'    C           LBZ     CLRCLO                        ;IF SO RESET ERROR FLAG
0208'   8C          C           GLO     RC                            ;WAS THE NON-HEX ENTRY
0209'   FB 0D       C           XRI     CR                            ;A CARRIAGE RETURN ?
020B'   C2 0219'    C           LBZ     CLRCLO                        ;IF SO RESET ERROR FLAG
```

```
 020E'   F8 10          C               LDI     10H                     ;OTHERWISE INDICATE
 0210'   BC             C               PHI     RC                      ;A NON-HEX ENTRY AND
 0211'   F8 08          C       EXPHXN: LDI     LOW     (SCRACH+3)      ;PREPARE TO RESTORE RA
 0213'   A7             C               PLO     R7
 0214'   47             C               LDA     R7                      ;GET OLD RA HI
 0215'   BA             C               PHI     RA                      ;PUT IT BACK
 0216'   47             C               LDA     R7                      ;GET OLD RA LO
 0217'   AA             C               PLO     RA                      ;PUT IT BACK
                        C               RETURN                          ;GO BACK TO SAIL
 0218'   D5             C+
                        C       ;
 0219'   F8 00          C       CLRCLO: LDI     00                      ;CLEAR ERROR FLAGS
 021B'   BC             C               PHI     RC
 021C'   C0 0211'       C               LBR     EXPHXN                  ;RETURN TO SAIL
                        C       ;
                                ;
                        C               INCLUDE TYPEC.MAC
                        C       ;       *************
                        C       ;       * TYPEC.MAC *
                        C       ;       *************
                        C       ;
                        C       ; -------------------------------------------
                        C       ; + CONVERT THE CONTENT OF RC TO ASCII AND TYPE +
                        C       ; -------------------------------------------
                        C       ;               (RC)
                        C       ;
                        C       ;This subroutine converts the hex contents of  .
                        C       ;the low half of RC to two ASCII characters, stores
                        C       ;the characters at "ASCII", and types them via SALTTY.
                        C       ;
 021F'                  C       TYPEC:: CALL    HTOA            ;CONVERT LOW BYTE
 021F'   D4             C+
 0220'   0085'          C+
 0222'   F8 07          C               LDI     LOW (SCRACH+2)  ;PREPARE TO STORE
 0224'   A7             C               PLO     R7              ;RESULT OF CONVERT
 0225'   E7             C               SEX     R7
 0226'   F8 7E          C               LDI     STOP            ;STORE TTY STOP
 0228'   73             C               STXD
 0229'   9C             C               GHI     RC              ;GET RESULT OF CONVERT
 022A'   73             C               STXD                    ;AND STORE IT
 022B'   8C             C               GLO     RC              ;GET READY TO CONVERT
 022C'   F6             C               SHR                     ;LOW ORDER BYTE
 022D'   F6             C               SHR
 022E'   F6             C               SHR
 022F'   F6             C               SHR
 0230'   AC             C               PLO     RC
                        C               CALL    HTOA            ;CONVERT HIGH ORDER
 0231'   D4             C+
 0232'   0085'          C+
 0234'   9C             C               GHI     RC             .;BYTE, GET RESULT
 0235'   57             C               STR     R7              ;AND STORE IT
                        C               CALL    SALTTY          ;TYPE THEM
 0236'   D4             C+
 0237'   00CB'          C+
 0239'   FF05           C               DW      SCRACH
                        C               RETURN
```

```
023B'   D5              C+
                        C       ;
                                ;
                        C               INCLUDE GET2HX.MAC
                        C       ;       ***************
                        C       ;       * GET2HX.MAC *
                        C       ;       ***************
                        C       ;
                        C       ; ------------------------------
                        C       ; + GET TWO FOUR DIGIT HEX NUMBERS +
                        C       ; ------------------------------
                        C       ;               (RA + RB + RC)
                        C       ;
                        C       ;This subroutine obtains two four digit hex numbers.
                        C       ;The first number is placed in RA, the second in RB.
                        C       ;
023C'                   C       GET2HX::                        ;THIS IS A PUBLIC
                        C               CALL    PHXIN           ;PROMPT FOR FIRST
023C'   D4              C+
023D'   01E6'           C+
023F'   03DD'           C               DW      FROM            ;NUMBER
0241'   9C              C               GHI     RC              ;TEST FOR UART ERRORS
0242'   CA 0252'        C               LBNZ    X2HEX           ;EXIT IF ERROR FOUND
0245'   9B              C               GHI     RB              ;PLACE FIRST NUMBER
0246'   BA              C               PHI     RA              ;IN REGISTER RA
0247'   8B              C               GLO     RB
0248'   AA              C               PLO     RA
                        C               CALL    PHXIN           ;PROMPT FOR SECOND
0249'   D4              C+
024A'   01E6'           C+
024C'   03EB'           C               DW      OVER            ;NUMBER
024E'   9C              C               GHI     RC              ;LOOK FOR UART ERRORS
024F'   CA 0252'        C               LBNZ    X2HEX           ;EXIT IF ERROR FOUND
0252'                   C       X2HEX:  RETURN                  ;EXIT
0252'   D5              C+
                        C       ;
                                ;
                        C               INCLUDE CALCRC.MAC
                        C       ;       ***************
                        C       ;       * CALCRC.MAC *
                        C       ;       ***************
                        C       ;
                        C       ; ---------------------------
                        C       ; + CALCULATE A NEW CRC VALUE +
                        C       ; ---------------------------
                        C       ;
                        C       ;This subroutine will calculate a new value CRC each
                        C       ;time it is called. The old value will be over
                        C       ;written, the address pointer ( RA ) will be
                        C       ;incremented, and the block counter ( RB ) will
                        C       ;be decremented
                        C       ;
0253'   F8 0B           C       CALCRC: LDI     LOW     (CRCHI) ;POINT TO CRC HI
0255'   AC              C               PLO     RC              ;USE RC AS THE POINTER
0256'   F8 FF           C               LDI     HIGH    (CRCHI)
0258'   BC              C               PHI     RC
```

```
0259'  F8 00     C          LDI     00          ;POINT TO A SCRATCH
025B'  A7        C          PLO     R7          ;LOCATION WITH GLOBAL
025C'  EC        C          SEX     RC          ;POINT TO CRC HI BYTE
025D'  4A        C          LDA     RA          ;GET MEMORY BYTE
025E'  F3        C          XOR                 ;XOR WITH MEMORY BYTE
025F'  57        C          STR     R7          ;SAVE RESULT
0260'  F6        C          SHR                 ;DIVIDE RESULT
0261'  F6        C          SHR                 ;BY 16
0262'  F6        C          SHR
0263'  F6        C          SHR
0264'  E7        C          SEX     R7          ;POINT TO RESULT
0265'  F3        C          XOR                 ;OF FIRST XOR AND XOR
0266'  57        C          STR     R7          ;WITH RESULT OF DIVIDE
0267'  FE        C          SHL                 ;MULTIPLY BY 16
0268'  FE        C          SHL
0269'  FE        C          SHL
026A'  FE        C          SHL
026B'  1C        C          INC     RC          ;POINT AT CRC LO BYTE
026C'  EC        C          SEX     RC
026D'  F3        C          XOR                 ;XOR WITH RESULT OF
026E'  2C        C          DEC     RC          ;MULTIPLY, AND STORE
026F'  5C        C          STR     RC          ;RESULT AT CRC HI BYTE
0270'  07        C          LDN     R7          ;GET RESULT OF SECOND
0271'  F6        C          SHR                 ;XOR, AND DIVIDE
0272'  F6        C          SHR                 ;IT BY 8
0273'  F6        C          SHR
0274'  F3        C          XOR                 ;XOR WITH CRC HI BYTE
0275'  5C        C          STR     RC          ;RESULT IS NEW CRC HI
0276'  07        C          LDN     R7          ;GET RESULT OF SECOND
0277'  FE        C          SHL                 ;XOR AND
0278'  FE        C          SHL                 ;MULTIPLY IT BY 32
0279'  FE        C          SHL
027A'  FE        C          SHL
027B'  FE        C          SHL
027C'  E7        C          SEX     R7          ;XOR THIS PRODUCT WITH
027D'  F3        C          XOR                 ;THE PRODUCT OF THE
027E'  1C        C          INC     RC          ;FIRST MULTIPLY
027F'  5C        C          STR     RC          ;RESULT IS NEW CRC LO
0280'  17        C          INC     R7          ;POINT AT SYSTEM FLAG
                 C          RETURN              ;EXIT
0281'  D5        C+
                 C    ;
                      ;
                 C          INCLUDE IM1CLK.MAC
                 C    ;     *************
                 C    ;     * M1CLK.MAC *
                 C    ;     *************
                 C    ;
                 C    ;--------------------------------
                 C    ;+ INCREMENT THE CLOCK BY ONE MINUTE +
                 C    ;--------------------------------
                 C    ;               (RA + RC)
                 C    ;
                 C    ;This subroutine will increment the software
                 C    ;clock by one minute. The year day will be
                 C    ;reset to 001 one day after year day 365,
                       C      ;i.e., leap year not allowed ! A second clock
                       C      ;which is always one minute ahead of the system
                       C      ;clock will also be incremented. If the "GO" flag
                       C      ;is set, the value at MINOW will be decremented.
                       C      ;If the new value at MINOW is equal to 1, Q will be
                       C      ;set indicating the start of a measurement sequence.
                       C      ;
                       C      ;Reserve seven locations in RAM to hold decimal
                       C      ;time code data of the system clock.
                       C      ;
FF10                   C      HD       EQU     GLOBAL+10H          ;HUNDREDS OF DAYS
FF16                   C      UM       EQU     GLOBAL+16H          ;UNITS OF MINUTES
                       C      ;
                       C      ;Reserve seven locations in RAM to hold decimal
                       C      ;time code data of the system clock time + 1 minute.
                       C      ;
FF29                   C      S1HD     EQU     GLOBAL+29H          ;HUNDREDS OF DAYS
FF2F                   C      S1UM     EQU     GLOBAL+2FH          ;UNITS OF MINUTES (+1)
                       C      ;
                       C      ;Reserve two locations in RAM to hold the "GO" flag.
                       C      ;This flag when set will be = AAAAH. The low half of the
                       C      ;"GO" flag is set in SCEDUL.MAC. A successful comparison
                       C      ;between the current time and the start time will set the
                       C      ;high half.
                       C      ;
FF43                   C      GOFLG    EQU     GLOBAL+43H          ;GO FLAG
                       C      ;
                       C      ;
0282'    7B            C      M1CLK::  SEQ                         ;USE Q AS A LOOP COUNTER
0283'    E7            C               SEX     R7                  ;USE R7 AS A POINTER
0284'    F8 16         C               LDI     LOW      (UM)       ;POINT AT SYSTEM TIME
0286'    A7            C      M2CLK:   PLO     R7                  ;START HERE FOR SECOND PASS
0287'    07            C               LDN     R7                  ;GET UNITS OF MINUTES
0288'    AA            C               PLO     RA
0289'    1A            C               INC     RA                  ;ADD 1 MINUTE
028A'    8A            C               GLO     RA                  ;GET NEW MIN. COUNT
028B'    FB 0A         C               XRI     0AH                 ;IS IT NOW 10 ?
028D'    CA 034E'      C               LBNZ    STRNEW              ;IF NOT, STORE NEW MIN.
                       C      ;
0290'    73            C               STXD                        ;STORE A 0 AT U.M.
0291'    07            C               LDN     R7                  ;GET TENS OF MINUTES
0292'    AA            C               PLO     RA
0293'    1A            C               INC     RA                  ;ADD 1 TO TEN MIN. CNT.
0294'    8A            C               GLO     RA                  ;GET NEW TEN MIN. CNT.
0295'    FB 06         C               XRI     06H                 ;IS IT NOW MINUTE 60 ?
0297'    CA 034E'      C               LBNZ    STRNEW              ;IF NOT STORE NEW T.M.C.
                       C      ;
029A'    73            C               STXD                        ;STORE 0 AT TM
029B'    07            C               LDN     R7                  ;GET UNITS OF HOURS
029C'    AA            C               PLO     RA
029D'    1A            C               INC     RA                  ;ADD 1 TO UNITS OF HRS.
029E'    8A            C               GLO     RA                  ;GET NEW U.H. COUNT
029F'    FB 0A         C               XRI     0AH                 ;IS IT NOW 10 ?
02A1'    C2 034A'      C               LBZ     INCTH               ;IF SO INC. T.H.
02A4'    8A            C               GLO     RA                  ;RESTORE NEW U.H. CNT.
02A5'    FB 04         C               XRI     04H                 ;IS IT NOW 4 ?
```

```
 02A7'   CA 034E'        C               LBNZ    STRNEW          ;IF NOT STORE NEW U.H.C.
                         C       ;
                         C       ;The units of hour counter is now 4. If the tens of hour
                         C       ;counter is now 2, both these counters will be reset, and
                         C       ;the units of days counter will be incremented.
                         C       ;
 02AA'   27              C               DEC     R7              ;POINT AT TENS OF HOURS
 02AB'   47              C               LDA     R7              ;GET TENS OF HOURS
 02AC'   FB 02           C               XRI     02H             ;IS IT NOW 2 ?
 02AE'   CA 034E'        C               LBNZ    STRNEW          ;IF NOT STORE A 4 AT UH
 02B1'   73              C               STXD                    ;ZERO TO UNITS OF HOURS
 02B2'   73              C               STXD                    ;ZERO TO TENS OF HOURS
                         C       ;
                         C       ;This loop will up date the days counter.
                         C       ;
 02B3'   F8 03           C               LDI     03H             ;SET LOOP COUNTER
 02B5'   AC              C               PLO     RC
 02B6'   07              C       UPDATE: LDN     R7              ;GET A DAY DIGIT
 02B7'   AA              C               PLO     RA
 02B8'   1A              C               INC     RA              ;ADD 1 DAY COUNT
 02B9'   8A              C               GLO     RA              ;GET NEW DAY COUNT
 02BA'   FB 0A           C               XRI     0AH             ;IS IT NOW 10 ?
 02BC'   CA 034E'        C               LBNZ    STRNEW          ;IF NOT STORE NEW DAY COUNT
 02BF'   73              C               STXD                    ;ZERO THIS DIGIT
 02C0'   2C              C               DEC     RC              ;DEC. LOOP COUNTER
 02C1'   8C              C               GLO     RC              ;IF THIS COUNTER IS
 02C2'   CA 02B6'        C               LBNZ    UPDATE          ;ZERO EXIT THE LOOP
                         C       ;
                         C       ;If it is year day 366, reset to day 001
                         C       ;
 02C5'   C9 0353'        C       LYPYR?: LBNQ    PS1HD           ;POINT AT S1HD SECOND PASS
 02C8'   F8 10           C               LDI     LOW     (HD)    ;OTHERWISE POINT AT HD
 02CA'   A7              C       LYPYR1: PLO     R7              ;POINT AT HUNDREDS OF DAYS
 02CB'   47              C               LDA     R7              ;IS IT DAY 3nn ?
 02CC'   FB 03           C               XRI     03H             ;IF NOT RETURN TO MAIN.
 02CE'   CA 02E4'        C               LBNZ    TSTQ            ;IF IT WAS DAY 3nn,
 02D1'   47              C               LDA     R7              ;IS IT DAY 36n ?
 02D2'   FB 06           C               XRI     06H             ;IF NOT RETURN TO MAIN.
 02D4'   CA 02E4'        C               LBNZ    TSTQ            ;IF IT WAS DAY 36n,
 02D7'   07              C               LDN     R7              ;IS IT DAY 366 ?
 02D8'   FB 06           C               XRI     06H             ;IF NOT RETURN TO MAIN
 02DA'   CA 02E4'        C               LBNZ    TSTQ            ;IF IT WAS DAY 366,
 02DD'   F8 01           C               LDI     01H             ;RESET DAYS TO 001
 02DF'   73              C               STXD
 02E0'   F8 00           C               LDI     00H
 02E2'   73              C               STXD
 02E3'   57              C               STR     R7              ;AND RETURN MAIN
                         C       ;
                         C       ;If Q is set this is the first time through the loop. Point
                         C       ;at time plus 1 minute, copy current time to this local, reset
                         C       ;Q and go through the loop a second time. If Q is not set, test
                         C       ;the condition of the "GO" flag.
                         C       ;
 02E4'   C9 0300'        C       TSTQ:   LBNQ    TSTGF           ;TEST Q AND IF SET
 02E7'   F8 10           C               LDI     LOW     (HD)    ;COPY CURRENT TIME
 02E9'   A7              C               PLO     R7              ;TO S1HD
```

```
02EA'  F8 29      C             LDI    LOW    (S1HD)
02EC'  AA         C             PLO    RA            ;USE RA AS A POINTER
02ED'  97         C             GHI    R7
02EE'  BA         C             PHI    RA
02EF'  F8 07      C             LDI    07H           ;USE RC AS A LOOP COUNTER
02F1'  AC         C             PLO    RC            ;INITIALLY SET TO 7
02F2'  47         C     CPYTIM: LDA    R7            ;GET A DIGIT OF CURRENT
02F3'  5A         C             STR    RA            ;TIME AND STORE IT
02F4'  1A         C             INC    RA            ;MOVE POINTER
02F5'  2C         C             DEC    RC            ;TEST COUNTER
02F6'  8C         C             GLO    RC            ;AND IF DONE
02F7'  CA 02F2'   C             LBNZ   CPYTIM        ;POINT
02FA'  F8 2F      C             LDI    LOW    (S1UM) ;AT TIME +1 MINUTE
02FC'  7A         C             REQ                  ;RESET LOOP COUNTER
02FD'  C0 0286'   C             LBR    M2CLK         ;GO THROUGH AGAIN
                  C     ;
                  C     ;Since Q was not set this is the second pass through
                  C     ;the loop. Decrement the measurement interval counter,
                  C     ;if and only if the "GO" flag is equal to AAAAH. If
                  C     ;after decrementing the measurement interval counter
                  C     ;its new value is 01H, request a measurement by setting
                  C     ;Q prior to exiting.
                  C     ;
0300'  F8 43      C     TSTGF:  LDI    LOW    (GOFLG) ;POINT AT GO FLAG
0302'  A7         C             PLO    R7
0303'  47         C             LDA    R7            ;AND IF NOT SET EXIT
0304'  FB AA      C             XRI    0AAH
0306'  CA 0328'   C             LBNZ   TSTIME        ;IF SET, TEST FOR
0309'  07         C             LDN    R7            ;START TIME
030A'  FB AA      C             XRI    0AAH
030C'  CA 0328'   C             LBNZ   TSTIME
030F'  F8 26      C             LDI    LOW    (MINOW) ;GET CURRENT INTERVAL
0311'  A7         C             PLO    R7            ;COUNT AND DECREMENT
0312'  47         C             LDA    R7
0313'  BA         C             PHI    RA
0314'  07         C             LDN    R7
0315'  AA         C             PLO    RA
0316'  2A         C             DEC    RA
0317'  8A         C             GLO    RA            ;SAVE NEW INTERVAL
0318'  73         C             STXD
0319'  9A         C             GHI    RA
031A'  57         C             STR    R7
031B'  CA 0358'   C             LBNZ   TSTICK        ;EXIT IF NEW INTERVAL IS
031E'  8A         C             GLO    RA            ;NOT EQUAL TO 1 MINUTE
031F'  FB 01      C             XRI    01H
0321'  CA 0358'   C             LBNZ   TSTICK
0324'  7B         C             SEQ                  ;OTHERWISE, SET Q FIRST
0325'  C0 0358'   C             LBR    TSTICK        ;THEN EXIT
                  C     ;
                  C     ;Compare current time +1 minute with start time. If they
                  C     ;are equal set the GO flag and request a branch to the
                  C     ;measurement sequence by exiting with Q set.
                  C     ;
0328'  F8 30      C     TSTIME: LDI    LOW    (DSHD) ;POINT AT START TIME
032A'  AA         C             PLO    RA            ;USING RA AS THE
032B'  97         C             GHI    R7            ;POINTER
```

```
032C'   BA          C           PHI     RA
032D'   F8 29       C           LDI     LOW     (S1HD)  ;POINT AT SYSTEM TIME
032F'   A7          C           PLO     R7              ;PLUS 1 MINUTE
0330'   E7          C           SEX     R7
0331'   8A          C   NXTXOR: GLO     RA              ;COMPARED ALL SEVEN ?
0332'   FB 37       C           XRI     LOW     (DSHD+7)
0334'   C2 0340'    C           LBZ     SGOFLG          ;IF SO SET GO FLAG
0337'   4A          C           LDA     RA              ;OTHERWISE, GET NEXT
0338'   F3          C           XOR                     ;AND COMPARE
0339'   CA 0358'    C           LBNZ    TSTICK          ;BRANCH TO TEST TICK
033C'   17          C           INC     R7              ;ON A MISMATCH, OTHERWISE
033D'   C0 0331'    C           LBR     NXTXOR          ;CONTINUE
0340'   F8 44       C   SGOFLG: LDI     LOW     (GOFLG+1)
0342'   A7          C           PLO     R7              ;SET HIGH HALF OF GO FLAG
0343'   F8 AA       C           LDI     0AAH
0345'   57          C           STR     R7
0346'   7B          C           SEQ                     ;REQUEST A MEASUREMENT
0347'   C0 0358'    C           LBR     TSTICK          ;EXIT
                    C   ;
034A'   73          C   INCTH:  STXD                    ;ZERO TO UNITS OF HOURS
034B'   07          C           LDN     R7              ;GET TENS OF HOURS
034C'   AA          C           PLO     RA
034D'   1A          C           INC     RA              ;ADD 1 TO TENS OF HOURS
                    C   ;
034E'   8A          C   STRNEW: GLO     RA              ;GET NEW COUNT
034F'   57          C           STR     R7              ;STORE IT
0350'   C0 02C5'    C           LBR     LYPYR?          ;RETURN TO MAIN
0353'   F8 29       C   PS1HD:  LDI     LOW     (S1HD)  ;SECOND TIME THROUGH
0355'   C0 02CA'    C           LBR     LYPYR1          ;POINT AT FAST CLOCK
                    C   ;
                    C   ;Depending on the state of the tick flag, this routine
                    C   ;will return to either the main program or the interrupt
                    C   ;service routine.
                    C   ;
0358'   F8 17       C   TSTICK: LDI     LOW     (TICK)  ;POINT AT TICK FLAG
035A'   A7          C           PLO     R7
035B'   07          C           LDN     R7              ;EXAMINE FLAG
035C'   C2 037A'    C           LBZ     ENTINT          ;IF NOT SET RETURN TO INTRPT.
                    C           RETURN                  ;OTHERWISE, RETURN TO MAIN
035F'   D5          C+
                        ;
                    C           INCLUDE IINTRPT.MAC
                    C   ;       **************
                    C   ;       * INTRPT.MAC *
                    C   ;       **************
                    C   ;
                    C   ;—————————————————————————————
                    C   ;+ THIS IS THE INTERRUPT SERVICE ROUTINE +
                    C   ;—————————————————————————————
                    C   ;
                    C   ;This routine handles interrupt requests. The only
                    C   ;interrupt which can occur in this system is the
                    C   ;one minute tick. The purpose of this routine is
                    C   ;simply to update the clock. Two exits are possible,
                    C   ;the normal exit, and the forced exit. A forced exit
                    C   ;occurs when upon returning from the routine which
```

```
                          C      ;advances the clock, Q is set indicating the start
                          C      ;of a measurement sequence.
                          C      ;
 0360'   E2               C      EXINT:  SEX     R2      ;RESTORE STACK POINTER AND
 0361'   70               C              RET             ;ENABLE FURTHER INTERRUPTS
                          C      ;
 0362'   22               C      INTRPT: DEC     R2      ;POINT TO A CLEAR LOCATION
 0363'   78               C              SAV             ;SAVE OLD X AND P
 0364'   22               C              DEC     R2      ;MOVE TO NEXT LOCATION
 0365'   73               C              STXD            ;SAVE ACCUMULATOR
 0366'   76               C              RSHR            ;MOVE DF TO MSB OF D
 0367'   73               C              STXD            ;SAVE DF
 0368'   87               C              GLO     R7      ;SAVE THE CONTENTS OF R7
 0369'   73               C              STXD
                          C      ;
                          C      ;At this point we have preserved enough of the
                          C      ;register data to safely test the tick flag and
                          C      ;exit if it is set.
                          C      ;
 036A'   F8 17            C              LDI     LOW     (TICK)
 036C'   A7               C              PLO     R7
 036D'   07               C              LDN     R7      ;GET TICK FLAG
 036E'   CA 038E'         C              LBNZ    XINTF   ;EXIT IF SET
                          C      ;
                          C      ;Tick flag was not set so continue saving registers
                          C      ;
 0371'   8A               C              GLO     RA      ;SAVE RA
 0372'   73               C              STXD
 0373'   9A               C              GHI     RA
 0374'   73               C              STXD
 0375'   8C               C              GLO     RC      ;SAVE RC.0
 0376'   73               C              STXD
                          C      ;
                          C      ;With these registers preserved the clock may
                          C      ;now be incremented.
                          C      ;
 0377'   C0 0282'         C              LBR     M1CLK   ;INCREMENT THE CLOCK
 037A'   C9 038A'         C      ENTINT: LBNQ    RSTRX   ;IF Q IS SET, LOAD R3
 037D'   F8 4B'           C              LDI     LOW     (MSRSEQ)
 037F'   A3               C              PLO     R3      ;WITH THE ADDRESS OF
 0380'   F8 0F'           C              LDI     HIGH    (MSRSEQ)
 0382'   B3               C              PHI     R3      ;MEASUREMENT SEQUENCE
 0383'   F8 FF            C              LDI     LOW     (STACK)
 0385'   A2               C              PLO     R2      ;RESTORE STACK POINTER
 0386'   F8 FF            C              LDI     HIGH    (STACK)
 0388'   B2               C              PHI     R2      ;AND
 0389'   D3               C              SEP     R3      ;EXIT INTERRUPT, OTHERWISE
 038A'   E2               C      RSTRX:  SEX     R2      ;RESTORE POINTER AND
 038B'   C0 039A'         C              LBR     RESTR   ;RESTORE ALL REGISTERS
                          C      ;
                          C      ;This is the fast interrupt exit
                          C      ;
 038E'   12               C      XINTF:  INC     R2      ;POINT TO OLD R7.0
 038F'   42               C      EXCON:  LDA     R2
 0390'   A7               C              PLO     R7      ;RESTORE R7
 0391'   42               C              LDA     R2
```

```
0392'   FE          C               SHL                 ;RESTORE DF
0393'   42          C               LDA     R2          ;RESTORE ACCUMULATOR
0394'   E1          C               SEX     R1          ;ENABLE INTERRUPT HARDWARE
0395'   65          C               OUT     CLRINT
0396'   00          C               DB      00
0397'   C0 0360'    C               LBR     EXINT       ;EXIT INTERRUPT ROUTINE
                    C       ;
                    C       ;This is the slow interrupt exit.
                    C       ;
039A'   12          C       RESTR:  INC     R2          ;POINT TO OLD RC.0
039B'   42          C               LDA     R2          ;GET RC.0
039C'   AC          C               PLO     RC          ;RC.0
039D'   42          C               LDA     R2          ;RESTORE RA
039E'   BA          C               PHI     RA
039F'   42          C               LDA     R2
03A0'   AA          C               PLO     RA
03A1'   C0 038F'    C               LBR     EXCON       ;CONTINUE RESTORING DATA
                    C       ;
                            ;
                    C               INCLUDE ISAIL.MAC
                    C       ;               **************************
                    C       ;               * SAIL DRIVER (SAIL.MAC) *
                    C       ;               **************************
                    C       ;
                    C       ;This module is designed to be a "KERNEL" around which
                    C       ;the operating system of any SAIL oriented instrument
                    C       ;may be based. The program expects the UART to be an
                    C       ;1854 and the CPU to be an 1802. The UART should be
                    C       ;located at I/O ports 6 and 7, and have its ES (bar)
                    C       ;input connected to a loop status indicator. The RCA
                    C       ;Standard Call and Return Technique (SCRT) is used.
                    C       ;
                    C       ;       1. Define the NAME of the SAIL device.
                    C       ;       2. Define the PROMPT character to be used.
                    C       ;       3. Change the HELP file as required.
                    C       ;
                    C       ;Define a few RAM locations.
                    C       ;
FF05                C       SCRACH  EQU     GLOBAL+5        ;A SCRATCH LOCATION
FF0B                C       CRCHI   EQU     SCRACH+6        ;CRC HI BYTE
FF0C                C       CRCLO   EQU     CRCHI+1         ;CRC LO BYTE
                    C       ;
                    C       ;Note that GLOBAL will always be address nn00 and
                    C       ;defines the start of a RAM page to be used by all
                    C       ;routines. The first location will usually contain
                    C       ;either the last character typed or the contents of
                    C       ;the UART status register. The second location is
                    C       ;reserved for the system error flag. Register R7
                    C       ;will always point to some GLOBAL location.
                    C       ;
                    C       ;*******************************************
                    C       ;* CUSTOMIZED FOR INTERROGATOR DATA LOGGER *
                    C       ;*******************************************
                    C       ;
03A4'   F8 17       C       CLKTIC: LDI     LOW     (TICK)
03A6'   A7          C               PLO     R7
```

```
03A7'   F8 01         C           LDI     01H             ;SET THE TICK FLAG
03A9'   57            C           STR     R7
                      C           CALL    M1CLK           ;ADVANCE THE CLOCK
03AA'   D4            C+
03AB'   0282'         C+
03AD'   C1 0F4B'      C           LBQ     MSRSEQ          ;IF TIME MEASURE, OTHERWISE
03B0'   F8 01         C   SAIL:   LDI     01H             ;POINT TO THE FLAG
03B2'   A7            C           PLO     R7              ;WORD AND
03B3'   F8 00         C           LDI     00H             ;RESET ALL BITS
03B5'   E7            C           SEX     R7
03B6'   73            C           STXD
03B7'   6E            C           INP     DATA            ;CLEAR UART DA BIT
03B8'   E3            C           SEX     R3              ;CONFIGURE UART
03B9'   67            C           OUT     STATUS
03BA'   12            C           DB      CONFIG
03BB'   F8 17         C           LDI     LOW    (TICK)
03BD'   A7            C           PLO     R7
03BE'   F8 00         C           LDI     00H             ;RESET TICK FLAG
03C0'   57            C           STR     R7
03C1'   C0 07BC'      C           LBR     ADDRS?          ;TEST FOR CLOSED LOOP
                      C   ;
03C4'   E3            C   EXIT:   SEX     R3              ;CLEAR INTERRUPT LATCH
03C5'   65            C           OUT     CLRINT
03C6'   00            C           DB      00H
03C7'   71            C           DIS                     ;DISABLE INTERRUPTS
03C8'   33            C           DB      33H
03C9'   C0 0F48'      C           LBR     MAIN            ;EXIT THIS MODULE
                      C   ;
                      C   ;The device NAME may be any combination of alpha-
                      C   ;numeric characters.
                      C   ;
03CC'   49 32 7E      C   NAME:   DB      "I2",STOP       ;DEVICE NAME
                      C   ;
007E                  C   STOP    EQU     "~"             ;MESSAGE TERMINATOR
                      C   ;
                      C   ;Pick a character to be used as an instrument PROMPT
                      C   ;
003A                  C   PROMPT  EQU     ":"             ;THIS IS THE PROMPT
                      C   ;
                      C   ;Define the hex equivalent of an ASCII carriage
                      C   ;return,line feed, space,nul, bell, and, etx character.
                      C   ;
000D                  C   CR      EQU     0DH
000A                  C   LF      EQU     0AH
0000                  C   NUL     EQU     00H
0020                  C   SPACE   EQU     20H
0003                  C   ETX     EQU     03H
0007                  C   BEL     EQU     07H
                      C   ;
                      C   ;The UART is configured for 7 data bits, even parity,
                      C   ;and 1 stop bit. This configuration may be modified
                      C   ;by changing the byte stored at CONFIG.
                      C   ;
0012                  C   CONFIG  EQU     12H     ;UART CONFIGURATION
                      C   ;
                      C   ;Define the I/O
```

```
                              C      ;
0001                          C      SELECT  EQU     01H     ;BANK SELECT FOR PROM
0002                          C      PWRRST  EQU     02H     ;POWER LATCH RESET
0003                          C      PING    EQU     03H     ;TRANSMIT A PING
0004                          C      STPCLK  EQU     04H     ;STOP THE REAL TIME CLOCK
0005                          C      CLRINT  EQU     05H     ;ACKNOWLEDGE THE INTERRUPT
0006                          C      DATA    EQU     06H     ;UART DATA, IN OR OUT
0007                          C      STATUS  EQU     07H     ;UART CONTROL OR STATUS
                              C      ;
                              C      ;Store a few often used messages
                              C      ;
03CF'   3B                    C      EOL:    DB      ";"
03D0'   0A 0D 7E              C      CRLF:   DB      LF,CR,STOP
03D3'   20                    C      SPSP:   DB      SPACE
03D4'   20 7E                 C      SP:     DB      SPACE,STOP
03D6'   0A 0D 20 7E           C      CRLFSP: DB      LF,CR,SPACE,STOP
03DA'   52 43 7E              C      RC$:    DB      "RC",STOP
03DD'   20 46 72 6F           C      FROM:   DB      " From ",ETX,STOP
03E1'   6D 20 03 7E           C
03E5'   20 54 6F 20           C      TO:     DB      " To ",ETX,STOP
03E9'   03 7E                 C
03EB'   20 4F 76 65           C      OVER:   DB      " Over ",ETX,STOP
03EF'   72 20 03 7E           C
03F3'   20 3D 20 7E           C      EQ$:    DB      " = ",STOP
03F7'   20 43 4C 45           C      CLEAR:  DB      " CLEAR",STOP
03FB'   41 52 7E              C
03FE'   20 4E 4F 54           C      NO:     DB      " NOT ALLOWED!!",STOP
0402'   20 41 4C 4C           C
0406'   4F 57 45 44           C
040A'   21 21 7E              C
040D'   4F 43 4B 7E           C      LOCK:   DB      "OCK",STOP
0411'   4E 4C 4F 43           C      UNLOCK: DB      "NLOCK",STOP
0415'   4B 7E                 C
0417'   44 4C 45 7E           C      DLE:    DB      "DLE",STOP
041B'   49 4E 47 7E           C      ING:    DB      "ING",STOP
041F'   20 4F 4B 7E           C      OK:     DB      " OK",STOP
0423'   20 20 4F 4B           C      OK?:    DB      "  OK (Y/N) ?  ",ETX,STOP
0427'   20 28 59 2F           C
042B'   4E 29 20 3F           C
042F'   20 20 03 7E           C
0433'   0A 0D 20 3A           C      PRMPT:  DB      LF,CR,SPACE,PROMPT,SPACE,ETX,STOP
0437'   20 03 7E              C
043A'   20 20 57 48           C      ERROR:  DB      SPACE,SPACE,"WHAT ?",BEL,STOP
043E'   41 54 20 3F           C
0442'   07 7E                 C
0444'   6F 76 65 7E           C      OVE:    DB      "ove",STOP
0448'   20 52 45 41           C      READY:  DB      " READY",STOP
044C'   44 59 7E              C
044F'   20 30 30 2E           C      SECS:   DB      SPACE,"00... ",STOP
0453'   2E 2E 20 7E           C
0457'   40 7E                 C      AT:     DB      "@",STOP
0459'   49 4D 45 7E           C      TIME:   DB      "IME",STOP
045D'   43 48 45 44           C      SCED:   DB      "CHEDULE",STOP
0461'   55 4C 45 7E           C
0465'   0D 0A 20 53           C      STDAY:  DB      CR,LF," Start on day = ",STOP
0469'   74 61 72 74           C
```

```
046D'   20 6F 6E 20    C
0471'   64 61 79 20    C
0475'   3D 20 7E       C
0478'   20 20 68 6F    C     STHOUR: DB      "  hour = ",STOP
047C'   75 72 20 3D    C
0480'   20 7E          C
0482'   20 20 6D 69    C     STMIN:  DB      "  minute = ",STOP
0486'   6E 75 74 65    C
048A'   20 3D 20 7E    C
048E'   0D 0A 20 4D    C     MEAINT: DB      CR,LF," Measurement interval,  minutes = ",STOP
0492'   65 61 73 75    C
0496'   72 65 6D 65    C
049A'   6E 74 20 69    C
049E'   6E 74 65 72    C
04A2'   76 61 6C 2C    C
04A6'   20 20 6D 69    C
04AA'   6E 75 74 65    C
04AE'   73 20 3D 20    C
04B2'   7E             C
04B3'   0D 0A 20 53    C     SCDMSG: DB      CR,LF," Scheduler is ",STOP
04B7'   63 68 65 64    C
04BB'   75 6C 65 72    C
04BF'   20 69 73 20    C
04C3'   7E             C
04C4'   41 43 54 49    C     ACTIVE: DB      "ACTIVE with ",STOP
04C8'   56 45 20 77    C
04CC'   69 74 68 20    C
04D0'   7E             C
04D1'   48 20 6D 69    C     MINREM: DB      "H minutes remaining to the next measurement."
04D5'   6E 75 74 65    C
04D9'   73 20 72 65    C
04DD'   6D 61 69 6E    C
04E1'   69 6E 67 20    C
04E5'   74 6F 20 74    C
04E9'   68 65 20 6E    C
04ED'   65 78 74 20    C
04F1'   6D 65 61 73    C
04F5'   75 72 65 6D    C
04F9'   65 6E 74 2E    C
04FD'   7E             C             DB      STOP
04FE'   49 44 4C 45    C     IDLE1:  DB      "IDLE.",STOP
0502'   2E 7E          C
0504'   0D 0A 20 50    C     PNTR:   DB      CR,LF," Pointer is at ",STOP
0508'   6F 69 6E 74    C
050C'   65 72 20 69    C
0510'   73 20 61 74    C
0514'   20 7E          C
0516'   0D 0A 20 53    C     NOTACT: DB      CR,LF," Scheduler was NOT active !",BEL,STOP
051A'   63 68 65 64    C
051E'   75 6C 65 72    C
0522'   20 77 61 73    C
0526'   20 4E 4F 54    C
052A'   20 61 63 74    C
052E'   69 76 65 20    C
0532'   21 07 7E       C
0535'   0D 0A 20 54    C     MIMIN:  DB      CR,LF," This interval must be greater than "
```

```
0539'   68 69 73 20    C
053D'   69 6E 74 65    C
0541'   72 76 61 6C    C
0545'   20 6D 75 73    C
0549'   74 20 62 65    C
054D'   20 67 72 65    C
0551'   61 74 65 72    C
0555'   20 74 68 61    C
0559'   6E 20          C
055B'   74 77 6F 20    C             DB       "two (2) minutes.",BEL,STOP
055F'   28 32 29 20    C
0563'   6D 69 6E 75    C
0567'   74 65 73 2E    C
056B'   07 7E          C
056D'   41 52 4D 45    C     ARMIDL: DB       "ARMED BUT NOT ACTIVE",STOP
0571'   44 20 42 55    C
0575'   54 20 4E 4F    C
0579'   54 20 41 43    C
057D'   54 49 56 45    C
0581'   7E             C
0582'   4E 4F 54 20    C     NOTARM: DB       "NOT ARMED",STOP
0586'   41 52 4D 45    C
058A'   44 7E          C
058C'   0A 20 20 41    C     SAT:    DB       LF,"  AT  ",STOP
0590'   54 20 20 7E    C
0594'   2A 7E          C     ASTK:   DB       "*",STOP
0596'   61 6D 20 54    C     RMTST:  DB       "am Test",STOP
059A'   65 73 74 7E    C
                       C     ;
                       C     ;This is the HELP file. It contains an explanation
                       C     ;of the common 1802 monitor functions accessible
                       C     ;via the sail loop. Special functions that apply
                       C     ;to the program in which this module is placed may
                       C     ;be added here. The monitor functions included are:
                       C     ;?M, !M, SP, !LOCK, !UNLOCK, and ?C
                       C     ;
059E'   0D 0A 0A       C     HELP:   DB       CR,LF,LF
05A1'   20 20 49 4E    C             DB       "  INTERROGATOR PROGRAM"
05A5'   54 45 52 52    C
05A9'   4F 47 41 54    C
05AD'   4F 52 20 50    C
05B1'   52 4F 47 52    C
05B5'   41 4D          C
05B7'   20 20 20 56    C             DB       "   Ver. 1.1  Jan. 1985"
05BB'   65 72 2E 20    C
05BF'   31 2E 31 20    C
05C3'   20 4A 61 6E    C
05C7'   2E 20 31 39    C
05CB'   38 35          C
05CD'   0D 0A 0A 0A    C             DB       CR,LF,LF,LF
05D1'   20 20 53 59    C             DB       "  SYSTEM COMMANDS",CR,LF,LF
05D5'   53 54 45 4D    C
05D9'   20 43 4F 4D    C
05DD'   4D 41 4E 44    C
05E1'   53 0D 0A 0A    C
05E5'   20 20 21 4D    C             DB       "  !Maaaa dddd"
```

```
05E9'   61 61 61 61     C
05ED'   20 64 64 64     C
05F1'   64              C
05F2'   20 20 20 20     C               DB      "       ","LOAD MEMORY"
05F6'   20 4C 4F 41     C
05FA'   44 20 4D 45     C
05FE'   4D 4F 52 59     C
0602'   0A0D            C               DW      0A0DH
0604'   20 20 3F 4D     C               DB      "  ?M"
0608'   20 20 20 20     C               DB      "                ","DISPLAY MEMORY"
060C'   20 20 20 20     C
0610'   20 20 20 20     C
0614'   20 20 44 49     C
0618'   53 50 4C 41     C
061C'   59 20 4D 45     C
0620'   4D 4F 52 59     C
0624'   0A0D            C               DW      0A0DH
0626'   20 20 24 50     C               DB      "  $Paaaa"
062A'   61 61 61 61     C
062E'   20 20 20 20     C               DB      "           ","RUN PROGRAM"
0632'   20 20 20 20     C
0636'   20 20 52 55     C
063A'   4E 20 50 52     C
063E'   4F 47 52 41     C
0642'   4D              C
0643'   0A0D            C               DW      0A0DH
0645'   20 20 3F 43     C               DB      "  ?C"
0649'   20 20 20 20     C               DB      "                ","CALCULATE CRC"
064D'   20 20 20 20     C
0651'   20 20 20 20     C
0655'   20 20 43 41     C
0659'   4C 43 55 4C     C
065D'   41 54 45 20     C
0661'   43 52 43        C
0664'   0A0D            C               DW      0A0DH
0666'   20 20 20 4D     C               DB      "   M"
066A'   20 20 20 20     C               DB      "                ","MOVE MEMORY"
066E'   20 20 20 20     C
0672'   20 20 20 20     C
0676'   20 20 4D 4F     C
067A'   56 45 20 4D     C
067E'   45 4D 4F 52     C
0682'   59              C
0683'   0A0D            C               DW      0A0DH
0685'   20 20 20 52     C               DB      "   R"
0689'   20 20 20 20     C               DB      "                ","TEST RAM"
068D'   20 20 20 20     C
0691'   20 20 20 20     C
0695'   20 20 54 45     C
0699'   53 54 20 52     C
069D'   41 4D           C
069F'   0A0D            C               DW      0A0DH
06A1'   20 20 3F 53     C               DB      "  ?S"
06A5'   20 20 20 20     C               DB      "                ","DISPLAY SCHEDULE"
06A9'   20 20 20 20     C
06AD'   20 20 20 20     C
```

```
06B1'   20 20 44 49     C
06B5'   53 50 4C 41     C
06B9'   59 20 53 43     C
06BD'   48 45 44 55     C
06C1'   4C 45           C
06C3'   0A0D            C               DW      0A0DH
06C5'   20 20 21 53     C               DB      "  !SCHEDULE"
06C9'   43 48 45 44     C
06CD'   55 4C 45        C
06D0'   20 20 20 20     C               DB      "       ","PROGRAM SCHEDULE"
06D4'   20 20 20 50     C
06D8'   52 4F 47 52     C
06DC'   41 4D 20 53     C
06E0'   43 48 45 44     C
06E4'   55 4C 45        C
06E7'   0A0D            C               DW      0A0DH
06E9'   20 20 21 54     C               DB      "  !TIME"
06ED'   49 4D 45        C
06F0'   20 20 20 20     C               DB      "           ","SET CLOCK"
06F4'   20 20 20 20     C
06F8'   20 20 20 53     C
06FC'   45 54 20 43     C
0700'   4C 4F 43 4B     C
0704'   0A0D            C               DW      0A0DH
0706'   20 20 3F 54     C               DB      "  ?T"
070A'   20 20 20 20     C               DB      "              ","DISPLAY TIME"
070E'   20 20 20 20     C
0712'   20 20 20 20     C
0716'   20 20 44 49     C
071A'   53 50 4C 41     C
071E'   59 20 54 49     C
0722'   4D 45           C
0724'   0A0D            C               DW      0A0DH
0726'   20 20 21 4C     C               DB      "  !LOCK","           "
072A'   4F 43 4B 20     C
072E'   20 20 20 20     C
0732'   20 20 20 20     C
0736'   20 20           C
0738'   50 52 4F 54     C               DB      "PROTECT MEMORY"
073C'   45 43 54 20     C
0740'   4D 45 4D 4F     C
0744'   52 59           C
0746'   0A0D            C               DW      0A0DH
0748'   20 20 21 55     C               DB      "  !UNLOCK","         "
074C'   4E 4C 4F 43     C
0750'   4B 20 20 20     C
0754'   20 20 20 20     C
0758'   20 20           C
075A'   55 4E 50 52     C               DB      "UNPROTECT MEMORY"
075E'   4F 54 45 43     C
0762'   54 20 4D 45     C
0766'   4D 4F 52 59     C
076A'   0A0D            C               DW      0A0DH
076C'   20 20 21 49     C               DB      "  !IDLE","           "
0770'   44 4C 45 20     C
0774'   20 20 20 20     C
```

```
 0778'   20 20 20 20    C
 077C'   20 20          C
 077E'   49 4E 48 49    C               DB      "INHIBIT SCHEDULER"
 0782'   42 49 54 20    C
 0786'   53 43 48 45    C
 078A'   44 55 4C 45    C
 078E'   52             C
 078F'   0A0D           C               DW      0A0DH
 0791'   20 20 21 50    C               DB      "  !PING","           "
 0795'   49 4E 47 20    C
 0799'   20 20 20 20    C
 079D'   20 20 20 20    C
 07A1'   20 20          C
 07A3'   54 52 41 4E    C               DB      "TRANSMIT A 10 mS PULSE",CR,LF,STOP
 07A7'   53 4D 49 54    C
 07AB'   20 41 20 31    C
 07AF'   30 20 6D 53    C
 07B3'   20 50 55 4C    C
 07B7'   53 45 0D 0A    C
 07BB'   7E             C
                        C       ;
                        C       ;If data is available, the loop is closed.
                        C       ;Enable interrupts which will take over the
                        C       ;function of incrementing the real time clock,
                        C       ;and look for the address sequence. (#NAME)
                        C       ;
 07BC'   E3             C       ADDRS?: SEX     R3
 07BD'   65             C               OUT     CLRINT          ;RESET INTERRUPT
 07BE'   00             C               DB      00H             ;HARDWARE
 07BF'   70             C               RET                     ;ENABLE INTERRUPTS
 07C0'   33             C               DB      33H
                        C               CHAR?                   ;RECEIVED A "#" ?
 07C1'   D4             C+
 07C2'   0145'          C+
 07C4'   9C             C+
 07C5'   CA 0939'       C+
 07C8'   8C             C+
                        C       ;
                        C       ;Since the # was received, set up to receive NAME
                        C       ;
 07C9'                  C       DEVICE: WORD?   NAME            ;LOOK FOR "NAME"
 07C9'   D4             C+
 07CA'   012B'          C+
 07CC'   03CC'          C+
 07CE'   9C             C+
 07CF'   CA 0939'       C+
 07D2'   8C             C+
 07D3'   CA 07BC'       C               LBNZ    ADDRS?          ;TRY AGAIN
 07D6'   C0 08D2'       C               LBR     IDENT           ;IDENTIFY INSTRUMENT
                        C       ;
                        C       ;AT THIS POINT THE INSTRUMENT IS CORRECTLY ADDRESSED
                        C       ;
                                ;
                        C               INCLUDE ISCMDS.MAC
                        C       ;       *************
                        C       ;       * SCMDS.MAC *
                        C     ;        **************
                        C     ; ______________________________
                        C     ; + H, ?C, ?M, !M, SP, M, R, !LOCK,+
                        C     ; +!S, ?S, !T, ?T, !IDLE,!PING    +
                        C     ; +!UNLOCK  COMMAND INTERPRETER   +
                        C     ; ______________________________
                        C     ;
                        C     ;Test the first character received after a correct
                        C     ;address sequence. It should be an H, M, R, ?, !, or $.
                        C     ;If it is not, generate an error message.
                        C     ;
07D9'   F8 17           C     CMDIN:  LDI     LOW     (TICK)
07DB'   A7              C             PLO     R7
07DC'   F8 00           C             LDI     00H     ;CLEAR TICK FLAG
07DE'   57              C             STR     R7
                        C             CHAR?           ;GET A CHARACTER
07DF'   D4              C+
07E0'   0145'           C+
07E2'   9C              C+
07E3'   CA 0939'        C+
07E6'   8C              C+
07E7'   FB 48           C             XRI     "H"     ;IS IT AN "H"
07E9'   C2 0905'        C             LBZ     HLPOUT  ;IF SO TYPE THE HELP MESSAGE
07EC'   8C              C             GLO     RC      ;OTHERWISE, RESTORE CHARACTER
07ED'   FB 3F           C             XRI     "?"     ;IS IT A "?" ?
07EF'   C2 080D'        C             LBZ     CorM    ;IF SO TEST NEXT FOR C OR M
07F2'   8C              C             GLO     RC      ;OTHERWISE, RESTORE CHARACTER
07F3'   FB 21           C             XRI     "!"     ;IS IT A "!" ?
07F5'   C2 082F'        C             LBZ     MorL    ;IF SO TEST NEXT FOR M OR LOCK
07F8'   8C              C             GLO     RC      ;OTHERWISE, RESTORE CHARACTER
07F9'   FB 24           C             XRI     "$"     ;IS IT A "$" ?
07FB'   C2 08B1'        C             LBZ     P?      ;IF SO TEST NEXT FOR P
07FE'   8C              C             GLO     RC      ;OTHERWISE, RESTORE CHARACTER
07FF'   FB 4D           C             XRI     "M"     ;IS IT AN "M" ?
0801'   C2 0ADF'        C             LBZ     MOVE    ;IF SO GOTO MOVE
0804'   8C              C             GLO     RC      ;OTHERWISE, RESTORE CHARACTER
0805'   FB 52           C             XRI     "R"     ;IS IT AN "R" ?
0807'   C2 0BD5'        C             LBZ     RAMTST  ;IF TEST RAM
                        C     ;
                        C     ;This ends the test for the standard sail commands.
                        C     ;Enter any additional command tests after this
                        C     ;comment.
                        C     ;
                        C     ; ***** ADDITIONAL COMMAND TESTS GO HERE *********
                        C     ;
080A'   C0 08F9'        C             LBR     ERROUT  ;NOT A RECOGNIZED COMMAND
                        C     ;
                        C     ;Determine the character which follows "?". It should
                        C     ;be either a "C", "M", "T" or an "S". If it is not, type
                        C     ;the error message and go to PRMOUT.
                        C     ;
080D'                   C     CorM:   CHAR?           ;GET THE NEXT CHARACTER
080D'   D4              C+
080E'   0145'           C+
0810'   9C              C+
0811'   CA 0939'        C+
0814'   8C          C+
0815'   FB 43       C           XRI     "C"     ;IS IT A "C" ?
0817'   C2 0A58'    C           LBZ     CRC     ;IF SO GO TO CRC
081A'   8C          C           GLO     RC      ;OTHERWISE, RESTORE CHARACTER
081B'   FB 4D       C           XRI     "M"     ;IS IT AN "M" ?
081D'   C2 094D'    C           LBZ     QUERRY  ;IF SO GO TO QUERRY MEMORY
0820'   8C          C           GLO     RC      ;OTHERWISE, RESTORE CHARACTER
0821'   FB 54       C           XRI     "T"     ;IS IT A "T"
0823'   C2 0B35'    C           LBZ     QUETIM  ;IF SO GO TO QUESTION TIME
0826'   8C          C           GLO     RC      ;OTHERWISE, RESTORE CHARACTER
0827'   FB 53       C           XRI     "S"     ;IS IT AN "S" ?
0829'   C2 0DD6'    C           LBZ     QRYSCED ;IF SO TYPE SCHEDULE
                    C   ;
                    C   ; **** ENTER ADDITIONAL "?" COMMANDS HERE ****
                    C   ;
082C'   C0 08F9'    C           LBR     ERROUT  ;IF NOT, TYPE THE ERROR MSG.
                    C   ;
                    C   ;Determine which characters follow the "!". They
                    C   ;should be either an "M", "LOCK", "UNLOCK", "I",
                    C   ;or "P";
082F'               C   MorL:   CHAR?           ;GET THE NEXT CHARACTER
082F'   D4          C+
0830'   0145'       C+
0832'   9C          C+
0833'   CA 0939'    C+
0836'   8C          C+
0837'   FB 4D       C           XRI     "M"     ;IS IT AN "M"
0839'   C2 09A4'    C           LBZ     LOAD    ;IF SO GO TO LOAD
                    C   ;
                    C   ;The command was not !M, so test for !LOCK, !UNLOCK
                    C   ;!IDLE, or !PING.
                    C   ;
083C'   8C          C           GLO     RC      ;RESTORE CHARACTER
083D'   FB 4C       C           XRI     "L"     ;IS IT AN "L" ?
083F'   CA 084F'    C           LBNZ    U?      ;IF SO LOOK FOR "OCK"
                    C           WORD?   LOCK    ;AND IF FOUND GOTO
0842'   D4          C+
0843'   012B'       C+
0845'   040D'       C+
0847'   9C          C+
0848'   CA 0939'    C+
084B'   8C          C+
084C'   C2 0911'    C           LBZ     CLOSE   ;CLOSE, OTHERWISE
084F'   8C          C   U?:     GLO     RC      ;RESTORE CHARACTER
0850'   FB 55       C           XRI     "U"     ;IS IT A "U" ?
0852'   CA 0862'    C           LBNZ    T?      ;IF SO LOOK FOR "NLOCK"
                    C           WORD?   UNLOCK  ;AND IF FOUND GOTO
0855'   D4          C+
0856'   012B'       C+
0858'   0411'       C+
085A'   9C          C+
085B'   CA 0939'    C+
085E'   8C          C+
085F'   C2 0925'    C           LBZ     OPEN    ;OPEN, OTHERWISE
0862'   8C          C   T?:     GLO     RC      ;RESTORE CHARACTER
0863'   FB 54       C           XRI     "T"     ;IS IT "T" ?
```

```
 0865'   CA 0875'       C               LBNZ     S?       ;IF SO LOOK FOR "IME"
                        C               WORD?    TIME     ;IF FOUND
 0868'   D4             C+
 0869'   012B'          C+
 086B'   0459'          C+
 086D'   9C             C+
 086E'   CA 0939'       C+
 0871'   8C             C+
 0872'   C2 0B90'       C               LBZ      LDTIM    ;SET THE CLOCK
 0875'   8C             C       S?:     GLO      RC       ;IS IT AN "S"
 0876'   FB 53          C               XRI      "S"      ;IF SO LOOK FOR
 0878'   CA 0888'       C               LBNZ     I?       ;"CHEDULE"
                        C               WORD?    SCED     ;AND IF FOUND
 087B'   D4             C+
 087C'   012B'          C+
 087E'   045D'          C+
 0880'   9C             C+
 0881'   CA 0939'       C+
 0884'   8C             C+
 0885'   C2 0CBE'       C               LBZ      LDSCED   ;LOAD THE SCHEDULE
 0888'   8C             C       I?:     GLO      RC       ;RESTORE CHARACTER
 0889'   FB 49          C               XRI      "I"      ;IS IT AN "I" ?
 088B'   CA 089B'       C               LBNZ     PN?      ;IF SO LOOK FOR DLE
                        C               WORD?    DLE      ;AND IF FOUND RESET
 088E'   D4             C+
 088F'   012B'          C+
 0891'   0417'          C+
 0893'   9C             C+
 0894'   CA 0939'       C+
 0897'   8C             C+
 0898'   C2 0F11'       C               LBZ      GFTOO    ;THE GO FLAG
 089B'   8C             C       PN?:    GLO      RC       ;OTHERWISE, IS IT
 089C'   FB 50          C               XRI      "P"      ;A "P" ?
 089E'   CA 08AE'       C               LBNZ     NOCMD    ;IF SO, LOOK FOR
                        C               WORD?    ING      ;"ING", AND IF FOUND
 08A1'   D4             C+
 08A2'   012B'          C+
 08A4'   041B'          C+
 08A6'   9C             C+
 08A7'   CA 0939'       C+
 08AA'   8C             C+
 08AB'   C2 0F37'       C               LBZ      TXMIT    ;SEND A PING
                        C       ;
                        C       ; **** ENTER ADDITIONAL "!" COMMANDS HERE ***
                        C       ;
 08AE'   C0 08F9'       C       NOCMD:  LBR      ERROUT   ;GOTO ERROUT
                        C       ;
                        C       ;Determine the character which follows the $,
                        C       ;it should be a "P".
                        C       ;
 08B1'                  C       P?:     CHAR?             ;GET THE NEXT CHARACTER
 08B1'   D4             C+
 08B2'   0145'          C+
 08B4'   9C             C+
 08B5'   CA 0939'       C+
 08B8'   8C             C+
08B9'   FB 50        C               XRI     "P"     ;IS IT A "P" ?
                     C       ;
                     C       ; **** ENTER ADDITIONAL "$" COMMANDS HERE ****
                     C       ;
08BB'   CA 08F9'     C               LBNZ    ERROUT  ;IF NOT GOTO ERROUT
                     C               GETFLG          ;IS THE SYSTEM OPEN ?
08BE'   F8 01        C+
08C0'   A7           C+
08C1'   07           C+
08C2'   F6           C               SHR
08C3'   C3 0AC0'     C               LBDF    RUN     ;IF IT IS GOTO RUN
08C6'                C       NORUN:  TYPMSG  NO      ;OTHERWISE, TYPE THE NO MSG.
08C6'   D4           C+
08C7'   00CB'        C+
08C9'   03FE'        C+
08CB'   9C           C+
08CC'   CA 0939'     C+
08CF'   C0 08ED'     C               LBR     PRMOUT  ;AND GOTO PRMOUT
                     C       ;
                     C       ;At this point all "command" tests have been made.
                     C       ;
                             ;
                     C               INCLUDE ISACTON.MAC
                     C       ;       **************
                     C       ;       * SACTON.MAC *
                     C       ;       **************
                     C       ;
                     C       ;This block of code defines the action to be taken
                     C       ;upon the receipt of standard SAIL commands
                     C       ;
08D2'                C       IDENT:  TYPMSG  CRLFSP          ;IDENTIFY BY TYPING
08D2'   D4           C+
08D3'   00CB'        C+
08D5'   03D6'        C+
08D7'   9C           C+
08D8'   CA 0939'     C+
                     C               TYPMSG  NAME            ;INSTRUMENT NAME AND
08DB'   D4           C+
08DC'   00CB'        C+
08DE'   03CC'        C+
08E0'   9C           C+
08E1'   CA 0939'     C+
                     C               TYPMSG  READY           ;THE WORD READY
08E4'   D4           C+
08E5'   00CB'        C+
08E7'   0448'        C+
08E9'   9C           C+
08EA'   CA 0939'     C+
                     C       ;
08ED'                C       PRMOUT: TYPMSG  PRMPT           ;TYPE PROMPT SEQUENCE
08ED'   D4           C+
08EE'   00CB'        C+
08F0'   0433'        C+
08F2'   9C           C+
08F3'   CA 0939'     C+
08F6'   C0 07D9'     C               LBR     CMDIN           ;GET ANOTHER COMMAND
```

```
                         C     ;
  08F9'                  C     ERROUT: TYPMSG  ERROR           ;TYPE ERROR SEQUENCE
  08F9'     D4           C+
  08FA'     00CB'        C+
  08FC'     043A'        C+
  08FE'     9C           C+
  08FF'     CA 0939'     C+
  0902'     C0 091A'     C             LBR     RSTFLG          ;GOTO PRMPT VIA RESET
                         C     ;
                         C     ;The response to the "H" command is to simply type
                         C     ;the message stored in the help file.
                         C     ;
  0905'                  C     HLPOUT: TYPMSG  HELP            ;TYPE THE HELP MESSAGE
  0905'     D4           C+
  0906'     00CB'        C+
  0908'     059E'        C+
  090A'     9C           C+
  090B'     CA 0939'     C+
  090E'     C0 08ED'     C             LBR     PRMOUT          ;GET NEXT COMMAND
                         C     ;
                         C     ;The response to a "LOCK" command is to reset the
                         C     ;system OPEN flag.
                         C     ;
  0911'                  C     CLOSE:  TYPMSG  OK              ;SAY OK THEN RESET FLAG
  0911'     D4           C+
  0912'     00CB'        C+
  0914'     041F'        C+
  0916'     9C           C+
  0917'     CA 0939'     C+
  091A'     F8 01        C     RSTFLG: LDI     01H             ;POINT AT SYSTEM FLAG
  091C'     A7           C             PLO     R7
  091D'     E7           C             SEX     R7
  091E'     F8 FE        C             LDI     0FEH            ;MASK ALL BUT OPEN BIT
  0920'     F2           C             AND                     ;RESET THIS BIT
  0921'     57           C             STR     R7              ;STORE FLAG
  0922'     C0 08ED'     C             LBR     PRMOUT          ;GET NEXT COMMAND
                         C     ;
                         C     ;The response to a "!UNLOCK" command is to set
                         C     ;the system OPEN flag.
                         C     ;
  0925'     F8 01        C     OPEN:   LDI     01H             ;POINT AT SYSTEM FLAG
  0927'     A7           C             PLO     R7
  0928'     E7           C             SEX     R7
  0929'     F8 01        C             LDI     01H             ;MASK ALL BUT OPEN BIT
  092B'     F1           C             OR                      ;SET THIS BIT
  092C'     57           C             STR     R7              ;STORE FLAG
                         C             TYPMSG  OK              ;TYPE "OK" AND
  092D'     D4           C+
  092E'     00CB'        C+
  0930'     041F'        C+
  0932'     9C           C+
  0933'     CA 0939'     C+
  0936'     C0 08ED'     C             LBR     PRMOUT          ;GET NEXT COMMAND
                         C     ;
                         C     ;Depending on the state of an error flag set
                         C     ;after reading the UART status, the program
```

```
                       C      ;will either branch to MAIN, un-address, or
                       C      ;output the error message.
                       C      ;
 0939'   9C            C      ERVEC:  GHI     RC              ;RECOVER STATUS FLAG
 093A'   FE            C              SHL                     ;LOOP OPEN ?
 093B'   C3 03C4'      C              LBDF    EXIT            ;IF SO EXIT
 093E'   FE            C              SHL                     ;RECEIVED A "#" ?
 093F'   C3 07C9'      C              LBDF    DEVICE          ;IF SO LOOK FOR NAME
 0942'   FE            C              SHL                     ;UART ERROR ?
 0943'   C3 03B0'      C              LBDF    SAIL            ;IF SO DE-ADDRESS
 0946'   FE            C              SHL                     ;OPERATOR ERROR ?
 0947'   C3 08F9'      C              LBDF    ERROUT          ;IF SO SEND ERROR MSG.
 094A'   C0 03C4'      C              LBR     EXIT            ;NONE OF ABOVE, EXIT
                       C      ;
                       C      ;The response to a "?M" is to type the contents of a
                       C      ;specified number of memory locations starting at
                       C      ;a specified address.
                       C      ;
 094D'                 C      QUERRY: CALL    GET2HX          ;GET START AND END
 094D'   D4            C+
 094E'   023C'         C+
                       C              ERROR?                  ;REACT TO ERRORS
 0950'   9C            C+
 0951'   CA 0939'      C+
                       C              TYPMSG  CRLF            ;TYPE A CR/LF
 0954'   D4            C+
 0955'   00CB'         C+
 0957'   03D0'         C+
 0959'   9C            C+
 095A'   CA 0939'      C+
 095D'   9A            C      TYPADD: GHI     RA              ;TYPE MSB OF ADDRESS
 095E'   AC            C              PLO     RC
                       C              CALL    TYPEC
 095F'   D4            C+
 0960'   021F'         C+
                       C              ERROR?                  ;REACT TO ERRORS
 0962'   9C            C+
 0963'   CA 0939'      C+
 0966'   8A            C              GLO     RA              ;TYPE LSB OF ADDRESS
 0967'   AC            C              PLO     RC
                       C              CALL    TYPEC
 0968'   D4            C+
 0969'   021F'         C+
                       C              ERROR?                  ;REACT TO ERRORS
 096B'   9C            C+
 096C'   CA 0939'      C+
 096F'                 C      SPOUT:  TYPMSG  SP              ;TYPE A SPACE
 096F'   D4            C+
 0970'   00CB'         C+
 0972'   03D4'         C+
 0974'   9C            C+
 0975'   CA 0939'      C+
 0978'   4A            C      BYTOUT: LDA     RA              ;GET A MEMORY BYTE
 0979'   AC            C              PLO     RC              ;TYPE IT
                       C              CALL    TYPEC
 097A'   D4            C+
097B'   021F'       C+
                    C               ERROR?                  ;REACT TO ERRORS
097D'   9C          C+
097E'   CA 0939'    C+
0981'   2B          C               DEC     RB              ;TEST FOR LAST LOCATION
0982'   9B          C               GHI     RB
0983'   CA 098A'    C               LBNZ    LETST
0986'   8B          C               GLO     RB              ;IF DONE PROMPT AND
0987'   C2 08ED'    C               LBZ     PRMOUT          ;GET NEXT COMMAND
098A'   8A          C       LETST:  GLO     RA              ;OTHERWISE, TEST FOR
098B'   FA 0F       C               ANI     0FH             ;END OF LINE
098D'   CA 099C'    C               LBNZ    TSTSP
                    C               TYPMSG  EOL             ;IF FOUND, TYPE ;
0990'   D4          C+
0991'   00CB'       C+
0993'   03CF'       C+
0995'   9C          C+
0996'   CA 0939'    C+
0999'   C0 095D'    C               LBR     TYPADD          ;CONTINUE
099C'   8A          C       TSTSP:  GLO     RA              ;NEXT LOCATION EVEN ?
099D'   F6          C               SHR                     ;IF SO, FIRST TYPE A
099E'   CB 096F'    C               LBNF    SPOUT           ;SPACE. OTHERWISE,
09A1'   C0 0978'    C               LBR     BYTOUT          ;TYPE NEXT MEMORY BYTE
                    C       ;
                    C       ;The response to a "!M" is to load memory with data
                    C       ;as it is input beginning at a specified address, and
                    C       ;continuing until a carriage return is encountered.
                    C       ;
09A4'               C       LOAD:   GETFLG          ;IS THE SYSTEM LOCKED ?
09A4'   F8 01       C+
09A6'   A7          C+
09A7'   07          C+
09A8'   F6          C               SHR             ;IF SO, TYPE THE ERROR
09A9'   C3 09AF'    C               LBDF    LDADD   ;MESSAGE THEN PROMPT
09AC'   C0 08C6'    C               LBR     NORUN   ;OTHERWISE,
09AF'   E7          C       LDADD:  SEX     R7      ;RESET THE SYSTEM
09B0'   F8 7E       C               LDI     7EH     ;LOCK FLAG AND THE
09B2'   F2          C               AND             ;COMPLETE BYTE FLAG
09B3'   57          C               STR     R7
                    C               CALL    GETHEX  ;GET START ADDRESS
09B4'   D4          C+
09B5'   016D'       C+
09B7'   9C          C               GHI     RC      ;TEST FOR UART ERRORS
09B8'   FA FE       C               ANI     0FEH    ;IF FOUND GOTO ERVEC
09BA'   CA 0939'    C               LBNZ    ERVEC
09BD'   8C          C               GLO     RC      ;WAS THE LAST CHARACTER
09BE'   FB 20       C               XRI     SPACE   ;A SPACE ?
09C0'   C2 09D2'    C               LBZ     NXTD    ;IF SO LOAD DATA
09C3'   8C          C               GLO     RC      ;WAS THE CHARACTER A
09C4'   FB 0D       C               XRI     CR      ;CARRIAGE RETURN ?
09C6'   C2 0A39'    C               LBZ     MODFLG  ;IF SO MODIFY COL. FLAG
09C9'   8C          C               GLO     RC      ;WAS THE CHARACTER A
09CA'   FB 0A       C               XRI     LF      ;LINE FEED ?
09CC'   C2 0A39'    C               LBZ     MODFLG  ;IF SO MODIFY COL. FLAG
09CF'   C0 08F9'    C               LBR     ERROUT  ;OTHERWISE, INDICATE AN ERROR
09D2'   F8 01       C       NXTD:   LDI     01H     ;POINT AT SYSTEM FLAG
```

```
 09D4'   A7          C            PLO     R7
 09D5'   E7          C            SEX     R7      ;RESET COL. FLAG BITS
 09D6'   F8 9F       C            LDI     9FH
 09D8'   F2          C            AND
 09D9'   57          C            STR     R7      ;STORE SYSTEM FLAG
                     C            CALL    INCHAR  ;GET NEXT CHARACTER
 09DA'   D4          C+
 09DB'   0145'       C+
                     C            ERROR?          ;REACT TO ERRORS
 09DD'   9C          C+
 09DE'   CA 0939'    C+
                     C            CALL    ATOH    ;CONVERT TO HEX
 09E1'   D4          C+
 09E2'   0058'       C+
 09E4'   9C          C            GHI     RC      ;ZERO FOR GOOD CONVERT
 09E5'   CA 0A06'    C            LBNZ    CRTST   ;REACT TO NON-HEX ENTRY
 09E8'   F8 04       C            LDI     04H     ;PREPARE TO ASSEMBLE
 09EA'   AA          C            PLO     RA      ;AN EIGHT BIT BYTE
 09EB'   8C          C    DSHFT:  GLO     RC      ;SHIFT HEX DIGIT FROM
 09EC'   FE          C            SHL             ;RA LOW TO RD HIGH
 09ED'   AC          C            PLO     RC
 09EE'   9A          C            GHI     RA
 09EF'   7E          C            RSHL
 09F0'   BA          C            PHI     RA
 09F1'   2A          C            DEC     RA
 09F2'   8A          C            GLO     RA
 09F3'   CA 09EB'    C            LBNZ    DSHFT
                     C            GETFLG          ;EIGHT BIT BYTE ASSEMBLED ?
 09F6'   F8 01       C+
 09F8'   A7          C+
 09F9'   07          C+
 09FA'   FE          C            SHL
 09FB'   C3 0A23'    C            LBDF    STORE   ;IF SO STORE IT, OTHERWISE
 09FE'   F8 80       C            LDI     80H     ;SET THE COMPLETE BYTE FLAG
 0A00'   E7          C            SEX     R7
 0A01'   F1          C            OR
 0A02'   57          C            STR     R7
 0A03'   C0 09D2'    C            LBR     NXTD    ;AND GET THE NEXT HEX DIGIT
 0A06'               C    CRTST:  GETFLG          ;THERE WAS A NON-HEX ENTRY
 0A06'   F8 01       C+
 0A08'   A7          C+
 0A09'   07          C+
 0A0A'   FE          C            SHL
 0A0B'   C3 08F9'    C            LBDF    ERROUT  ;IF THE BYTE WAS NOT COMPLETE
 0A0E'   8C          C            GLO     RC      ;THIS IS AN ERROR. IF THE BYTE
 0A0F'   FB 0D       C            XRI     CR      ;WAS COMPLETE AND THE ENTRY
 0A11'   C2 08ED'    C            LBZ     PRMOUT  ;WAS A CARRIAGE RETURN, EXIT
 0A14'   8C          C            GLO     RC      ;WAS THE ENTRY
 0A15'   FB 20       C            XRI     SPACE   ;A SPACE ?
 0A17'   C2 09D2'    C            LBZ     NXTD    ;IF SO CONTINUE
 0A1A'   8C          C            GLO     RC      ;WAS THE ENTRY A ";" ?
 0A1B'   FB 3B       C            XRI     ";"     ;IF SO SET THE COL. FLAG
 0A1D'   C2 0A2E'    C            LBZ     COLSET  ;AND CONTINUE. OTHERWISE
 0A20'   C0 08F9'    C            LBR     ERROUT  ;INDICATE AN ERROR AND EXIT
 0A23'   9A          C    STORE:  GHI     RA      ;GET THE BYTE TO BE STORED
 0A24'   5B          C            STR     RB      ;STORE IT
```

```
0A25'  F8 7E       C               LDI    7EH     ;RESET THE COMPLETE
0A27'  E7          C               SEX    R7      ;BYTE FLAG
0A28'  F2          C               AND
0A29'  57          C               STR    R7
0A2A'  1B          C               INC    RB      ;INCREMENT ADDRESS POINTER
0A2B'  C0 09D2'    C               LBR    NXTD    ;GET THE NEXT BYTE
0A2E'  F8 01       C       COLSET: LDI    01H     ;POINT AT SYSTEM FLAG
0A30'  A7          C               PLO    R7
0A31'  E7          C               SEX    R7
0A32'  F8 60       C               LDI    60H     ;INDICATE RECEPTION OF
0A34'  F1          C               OR             ;";" BY SETTING COL. FLAG
0A35'  57          C               STR    R7      ;SET THE FLAG
0A36'  C0 09AF'    C               LBR    LDADD   ;GET NEXT ADDRESS
0A39'              C       MODFLG: GETFLG         ;IF COL. FLAG IS SET
0A39'  F8 01       C+
0A3B'  A7          C+
0A3C'  07          C+
0A3D'  E7          C               SEX    R7      ;MODIFY IT TO REFLECT THE
0A3E'  FE          C               SHL            ;RECEPTION OF EITHER A
0A3F'  FE          C               SHL            ;CARRIAGE RETURN OR LINE FEED
0A40'  C3 0A4A'    C               LBDF   NOCR
0A43'  FE          C               SHL
0A44'  C3 0A51'    C               LBDF   NOLF    ;IF THIS FLAG WAS NOT SET
0A47'  C0 08F9'    C               LBR    ERROUT  ;THERE IS AN ERROR
0A4A'  F8 BF       C       NOCR:   LDI    0BFH    ;RESET CARRIAGE RETURN BIT
0A4C'  F2          C               AND
0A4D'  57          C               STR    R7      ;STORE MODIFIED FLAG
0A4E'  C0 09AF'    C               LBR    LDADD   ;GET NEW ADDRESS
0A51'  F8 DF       C       NOLF:   LDI    0DFH    ;RESET THE LINE FEED BIT
0A53'  F2          C               AND
0A54'  57          C               STR    R7      ;STORE MODIFIED FLAG
0A55'  C0 09AF'    C               LBR    LDADD   ;GET NEW ADDRESS
                   C
                   C       ;
                   C       ;The response to a ?C is to calculate the CRC of a
                   C       ;specified block of memory, beginning at a specified
                   C       ;location. An erased block will cause the CLEAR
                   C       ;to be typed.
                   C       ;
0A58'              C       CRC:    TYPMSG RC$             ;TYPE RC
0A58'  D4          C+
0A59'  00CB'       C+
0A5B'  03DA'       C+
0A5D'  9C          C+
0A5E'  CA 0939'    C+
0A61'  F8 0C       C               LDI    LOW     (CRCLO) ;SET RA TO POINT AT
0A63'  AA          C               PLO    RA              ;THE 16 BIT CRC
0A64'  F8 FF       C               LDI    HIGH    (CRCLO) ;CONSTANT
0A66'  BA          C               PHI    RA
0A67'  F8 00       C               LDI    00H
0A69'  EA          C               SEX    RA              ;SET THE CRC CONSTANT
0A6A'  73          C               STXD                   ;TO ZERO
0A6B'  5A          C               STR    RA
                   C               CALL   GET2HX          ;GET THE START ADDRESS
0A6C'  D4          C+
0A6D'  023C'       C+
                        C               ERROR?                      ;AND BLOCK SIZE
 0A6F'   9C             C+
 0A70'   CA 0939'       C+
 0A73'   F8 01          C               LDI     01H                 ;POINT TO SYSTEM FLAG
 0A75'   A7             C               PLO     R7
 0A76'   E7             C               SEX     R7
 0A77'   F8 80          C               LDI     80H                 ;SET THE CLEAR
-0A79'   F1             C               OR                          ;MEMORY FLAG
 0A7A'   57             C               STR     R7
 0A7B'   EA             C       CLOOP:  SEX     RA                  ;TEST A BYTE OF
 0A7C'   F8 FF          C               LDI     0FFH                ;MEMORY FOR THE
 0A7E'   F3             C               XOR                         ;CLEAR CONDITION
 0A7F'   C2 0A87'       C               LBZ     CALLCC              ;IF CLEAR, CALCULATE
 0A82'   E7             C               SEX     R7                  ;OTHERWISE, RESET
 0A83'   F8 7F          C               LDI     7FH                 ;CLEAR MEMORY FLAG
 0A85'   F2             C               AND
 0A86'   57             C               STR     R7
 0A87'                  C       CALLCC: CALL    CALCRC              ;CALCULATE CRC
 0A87'   D4             C+
 0A88'   0253'          C+
 0A8A'   2B             C               DEC     RB                  ;DECREMENT BYTE COUNT
 0A8B'   8B             C               GLO     RB                  ;TEST FOR CRC
 0A8C'   CA 0A7B'       C               LBNZ    CLOOP               ;CALCULATION COMPLETE
 0A8F'   9B             C               GHI     RB                  ;IF SO TEST FOR
 0A90'   CA 0A7B'       C               LBNZ    CLOOP               ;CLEAR MEMORY
 0A93'   07             C               LDN     R7                  ;GET SYSTEM FLAG
 0A94'   FE             C               SHL                         ;LOOK AT CLEAR MEMORY
 0A95'   CB 0AA4'       C               LBNF    CRCOUT              ;IF SET, TYPE THE
                        C               TYPMSG  CLEAR               ;CLEAR MESSAGE AND
 0A98'   D4             C+
 0A99'   00CB'          C+
 0A9B'   03F7'          C+
 0A9D'   9C             C+
 0A9E'   CA 0939'       C+
 0AA1'   C0 08ED'       C               LBR     PRMOUT              ;GET NEXT COMMAND
 0AA4'                  C       CRCOUT: TYPMSG  EQS                 ;TYPE =
 0AA4'   D4             C+
 0AA5'   00CB'          C+
 0AA7'   03F3'          C+
 0AA9'   9C             C+
 0AAA'   CA 0939'       C+
 0AAD'   F8 0B          C               LDI     LOW     (CRCHI) ;OTHERWISE, POINT
 0AAF'   AA             C               PLO     RA                  ;TO FINAL CRC
 0AB0'   F8 FF          C               LDI     HIGH    (CRCHI) ;CONSTANT AND TYPE IT
 0AB2'   BA             C               PHI     RA
 0AB3'   4A             C               LDA     RA                  ;GET HI HALF OF CRC
 0AB4'   AC             C               PLO     RC
                        C               CALL    TYPEC               ;TYPE IT
 0AB5'   D4             C+
 0AB6'   021F'          C+
 0AB8'   0A             C               LDN     RA                  ;GET LO HALF OF CRC
 0AB9'   AC             C               PLO     RC
                        C               CALL    TYPEC               ;TYPE IT
 0ABA'   D4             C+
 0ABB'   021F'          C+
 0ABD'   C0 08ED'       C               LBR     PRMOUT              ;GET NEXT COMMAND
```

```
                        C     ;
                        C     ;The response to a $P command is to run a program
                        C     ;beginning at a specified address. Prior to executing
                        C     ;this command, the X and P registers will be set to R0
                        C     ;
0AC0'   F8 01           C     RUN:    LDI     01H             ;RESET THE SYSTEM
0AC2'   A7              C             PLO     R7              ;LOCK FLAG
0AC3'   F8 FE           C             LDI     0FEH
0AC5'   F2              C             AND
0AC6'   57              C             STR     R7
                        C             CALL    GETHEX          ;GET START ADDRESS
0AC7'   D4              C+
0AC8'   016D'           C+
0ACA'   9C              C             GHI     RC              ;REACT TO UART ERRORS
0ACB'   FA FE           C             ANI     0FEH            ;MASK NON-HEX FLAG
0ACD'   CA 08F9'        C             LBNZ    ERROUT
0AD0'   8C              C             GLO     RC              ;LOOK FOR
0AD1'   FB 0D           C             XRI     CR              ;CARRIAGE RETURN
0AD3'   CA 08F9'        C             LBNZ    ERROUT          ;ERROR IF NOT FOUND
0AD6'   8B              C             GLO     RB              ;TRANSFER RUN ADDRESS
0AD7'   A0              C             PLO     R0              ;TO R0
0AD8'   9B              C             GHI     RB
0AD9'   B0              C             PHI     R0
0ADA'   E0              C             SEX     R0              ;SET X TO R0
0ADB'   D0              C             SEP     R0              ;RUN THE PROGRAM
0ADC'   C0 08ED'        C             LBR     PRMOUT          ;R3 LEFT POINTING HERE
                        C     ;
                        C     ;The response to an M typed as the command is to move
                        C     ;a block of memory from a specified location to a
                        C     ;specified location over a given length.
                        C
0ADF'                   C     MOVE:   GETFLG                  ;IS THE SYSTEM LOCKED ?
0ADF'   F8 01           C+
0AE1'   A7              C+
0AE2'   07              C+
0AE3'   F6              C             SHR                     ;IF SO, TYPE THE ERROR
0AE4'   C3 0AEA'        C             LBDF    SPEC            ;MESSAGE THEN PROMPT
0AE7'   C0 08C6'        C             LBR     NORUN           ;FOR NEXT COMMAND
                        C     ;
0AEA'   E7              C     SPEC:   SEX     R7              ;OTHERWISE, RESET THE
0AEB'   F8 FE           C             LDI     0FEH            ;LOCK FLAG
0AED'   F2              C             AND
0AEE'   57              C             STR     R7
                        C     ;
                        C             TYPMSG  OVE             ;PROMPT FOR SOURCE
0AEF'   D4              C+
0AF0'   00CB'           C+
0AF2'   0444'           C+
0AF4'   9C              C+
0AF5'   CA 0939'        C+
                        C             CALL    PHXIN           ;GET SOURCE ADDRESS
0AF8'   D4              C+
0AF9'   01E6'           C+
0AFB'   03DD'           C             DW      FROM
                        C             ERROR?                  ;LOOK FOR ERRORS
0AFD'   9C              C+
0AFE'   CA 0939'     C+
                     C     ;
0B01'   8B           C           GLO     RB          ;PLACE SOURCE ADDRESS
0B02'   AA           C           PLO     RA          ;IN RA
0B03'   9B           C           GHI     RB
0B04'   BA           C           PHI     RA
                     C     ;
                     C           CALL    PHXIN       ;GET DESTINATION ADDRESS
0B05'   D4           C+
0B06'   01E6'        C+
0B08'   03E5'        C           DW      TO
                     C           ERROR?              ;LOOK FOR ERRORS
0B0A'   9C           C+
0B0B'   CA 0939'     C+
                     C     ;
0B0E'   8B           C           GLO     RB          ;PLACE DESTINATION ADDRESS
0B0F'   AD           C           PLO     RD          ;IN RD
0B10'   9B           C           GHI     RB
0B11'   BD           C           PHI     RD
                     C     ;
                     C           CALL    PHXIN       ;GET BLOCK SIZE
0B12'   D4           C+
0B13'   01E6'        C+
0B15'   03EB'        C           DW      OVER
                     C           ERROR?              ;LOOK FOR ERRORS
0B17'   9C           C+
0B18'   CA 0939'     C+
                     C           CALL    ASKOK       ;ASK FINAL PERMISSION
0B1B'   D4           C+
0B1C'   0113'        C+
                     C           ERROR?              ;LOOK FOR UART ERRORS
0B1E'   9C           C+
0B1F'   CA 0939'     C+
0B22'   8C           C           GLO     RC          ;GET ANSWER
0B23'   CA 08ED'     C           LBNZ    PRMOUT      ;IF NOT YES EXIT
                     C     ;
0B26'   4A           C     MOVIT: LDA    RA          ;GET A BYTE
0B27'   5D           C           STR     RD          ;STORE IT
0B28'   1D           C           INC     RD          ;MOVE TO NEXT DESTINATION
0B29'   2B           C           DEC     RB          ;COUNT STORE OPERATION
0B2A'   9B           C           GHI     RB          ;TEST FOR BLOCK END
0B2B'   CA 0B26'     C           LBNZ    MOVIT       ;IF NOT AT END
0B2E'   8B           C           GLO     RB          ;CONTINUE, OTHERWISE
0B2F'   CA 0B26'     C           LBNZ    MOVIT
0B32'   C0 08ED'     C           LBR     PRMOUT      ;GET NEXT COMMAND
                     C     ;
                           ;
                     C           INCLUDE ITIME.MAC
                     C     ;
                     C     ;             *************
                     C     ;             * ITIME.MAC *
                     C     ;             *************
                     C     ;
                     C     ;  ________________________________
                     C     ; + SET AND READ THE SOFTWARE CLOCK +
                     C     ;  ________________________________
```

```
                         C       ;
                         C       ;The response to a ?T command is to first disable
                         C       ;further interrupts, then advance the clock. This
                         C       ;"future" time is then typed and a flag is set for
                         C       ;the interrupt service routine prior to re-enabling
                         C       ;interrupts and executing an idle instruction.
                         C       ;This flag will cause the interrupt routine to quickly
                         C       ;exit rather than advancing the clock. The interrupt
                         C       ;re-activates the system, and an @ is typed as an
                         C       ;immediate byte. This is the time tick.
                         C       ;
                         C       ;Define a byte of ram to be used as a flag word.
                         C       ;Define the start of ASCII time message.
                         C       ;
FF17                     C       TICK    EQU     GLOBAL+17H          ;TICK FLAG
FF18                     C       NXTM    EQU     TICK+01H            ;ASCII HD
                         C       ;
0B35'   E3               C       QUETIM: SEX     R3                  ;DISABLE INTERRUPTS
0B36'   71               C               DIS
0B37'   33               C               DB      33H
                         C       ;
                         C       ;Advance clock by one minute and convert this "future"
                         C       ;time to an ASCII message string.
                         C       ;
0B38'   F8 17            C               LDI     LOW     (TICK)      ;POINT TO TICK FLAG
0B3A'   A7               C               PLO     R7                  ;USING REGISTER R7
0B3B'   F8 01            C               LDI     01H                 ;SET TICK FLAG
0B3D'   57               C               STR     R7
                         C               CALL    M1CLK               ;ADVANCE TIME BY 1 MIN.
0B3E'   D4               C+
0B3F'   0282'            C+
0B41'   F8 18            C               LDI     LOW     (TICK+1)
0B43'   AA               C               PLO     RA
0B44'   97               C               GHI     R7                  ;USING RA, POINT TO
0B45'   BA               C               PHI     RA                  ;ASCII HUNDREDS OF DAYS
0B46'   F8 10            C               LDI     LOW     (HD)        ;POINT AT "FUTURE" TIME
0B48'   A7               C               PLO     R7                  ;USING R7
                         C               CALL    DTOA                ;CONVERT DAYS TO ASCII
0B49'   D4               C+
0B4A'   0095'            C+
0B4C'   03               C               DB      03H
0B4D'   F8 20            C               LDI     SPACE               ;STORE A SPACE
0B4F'   5A               C               STR     RA
0B50'   1A               C               INC     RA
                         C               CALL    DTOA                ;CONVERT HOURS TO ASCII
0B51'   D4               C+
0B52'   0095'            C+
0B54'   02               C               DB      02H
0B55'   F8 3A            C               LDI     ":"                 ;STORE A SEMI COLON
0B57'   5A               C               STR     RA
0B58'   1A               C               INC     RA
                         C               CALL    DTOA                ;CONVERT MINS. TO ASCII
0B59'   D4               C+
0B5A'   0095'            C+
0B5C'   02               C               DB      02H
0B5D'   F8 7E            C               LDI     STOP                ;STORE A STOP CHARACTER
```

```
0B5F'    5A            C            STR      RA
                       C            TYPMSG   SPSP              ;TYPE TWO SPACES
0B60'    D4            C+
0B61'    00CB'         C+
0B63'    03D3'         C+
0B65'    9C            C+
0B66'    CA 0939'      C+
                       C            TYPMSG   NXTM              ;OUTPUT TIME AT NEXT @
0B69'    D4            C+
0B6A'    00CB'         C+
0B6C'    FF18          C+
0B6E'    9C            C+
0B6F'    CA 0939'      C+
                       C            TYPMSG   SECS
0B72'    D4            C+
0B73'    00CB'         C+
0B75'    044F'         C+
0B77'    9C            C+
0B78'    CA 0939'      C+
0B7B'    E3            C            SEX      R3                ;RESET INTERRUPT HARDWARE
0B7C'    65            C            OUT      CLRINT
0B7D'    00            C            DB       00H
0B7E'    70            C            RET                        ;ENABLE INTERRUPTS
0B7F'    33            C            DB       33H
                       C    ;
0B80'    00            C            IDL                        ;GO TO SLEEP
                       C    ;
                       C            TYPMSG   AT                ;SEND THE @
0B81'    D4            C+
0B82'    00CB'         C+
0B84'    0457'         C+
0B86'    9C            C+
0B87'    CA 0939'      C+
0B8A'    C1 0F4B'      C            LBQ      MSRSEQ            ;MAKE A MEASUREMENT OR
0B8D'    C0 08ED'      C            LBR      PRMOUT            ;GET THE NEXT COMMAND
                       C    ;
                       C    ;This is the response to a !T command. The action taken is
                       C    ;to stop the clock, disable interrupts, and input time code.
                       C    ;Once the time code is input its ASCII bias is removed, then
                       C    ;the resulting digits are stored in RAM starting at "HD". When
                       C    ;nine digits have been input, (seconds must be 00) the system
                       C    ;waits for the @. Upon receiving an @, the clock is allowed to
                       C    ;run, interrupts are enabled, and a branch to CMDIN is executed.
                       C    ;
0B90'    E3            C    LDTIM:  SEX      R3                ;DISABLE INTERRUPTS
0B91'    71            C            DIS
0B92'    33            C            DB       33H
0B93'    F8 10         C            LDI      LOW      (HD)     ;POINT AT TIME DATA
0B95'    AA            C            PLO      RA
0B96'    F8 FF         C            LDI      HIGH     (HD)
0B98'    BA            C            PHI      RA
0B99'    F8 07         C            LDI      07H               ;SET THE DIGIT COUNTER
0B9B'    AD            C            PLO      RD                ;FOR SEVEN DIGITS
0B9C'    E3            C    INTIM:  SEX      R3                ;STOP THE CLOCK
0B9D'    64            C            OUT      STPCLK
0B9E'    00            C            DB       00H
```

```
                        C              CALL    INCHAR          ;GET A CHARACTER
 0B9F'    D4            C+
 0BA0'    0145'         C+
 0BA2'    9C            C              GHI     RC              ;MASK "#" BIT
 0BA3'    FA BF         C              ANI     0BFH            ;LOOK FOR UART ERRORS
 0BA5'    BC            C              PHI     RC              ;BRANCH TO ERVEC IF
 0BA6'    CA 0939'      C              LBNZ    ERVEC           ;FOUND, OTHERWISE
                        C              CALL    ATOH            ;IS IT A HEX NUMBER ?
 0BA9'    D4            C+
 0BAA'    0058'         C+
 0BAC'    9C            C              GHI     RC              ;IF NOT, GET ANOTHER
 0BAD'    CA 0B9C'      C              LBNZ    INTIM           ;NUMBER.
 0BB0'    8C            C              GLO     RC
 0BB1'    FF A0         C              SMI     0A0H            ;IS IT A DECIMAL NUMBER ?
 0BB3'    C3 0B9C'      C              LBDF    INTIM           ;IF NOT, GET ANOTHER
 0BB6'    8C            C              GLO     RC              ;NUMBER. IF IT WAS A
 0BB7'    F6            C              SHR                     ;DECIMAL NUMBER, MOVE
 0BB8'    F6            C              SHR                     ;IT TO THE LEAST
 0BB9'    F6            C              SHR                     ;SIGNIFICANT HALF OF
 0BBA'    F6            C              SHR                     ;THE ACCUMULATOR AND
 0BBB'    5A            C              STR     RA              ;STORE IT.
 0BBC'    1A            C              INC     RA              ;POINT TO NEXT LOCATION
 0BBD'    2D            C              DEC     RD              ;COUNT THE OPERATION
 0BBE'    8D            C              GLO     RD              ;ENTERED SEVEN DIGITS YET ?
 0BBF'    CA 0B9C'      C              LBNZ    INTIM           ;IF NOT GET NEXT NUMBER
 0BC2'    E3            C       AT?:   SEX     R3              ;OTHERWISE, STOP THE CLOCK
 0BC3'    64            C              OUT     STPCLK          ;AGAIN AND BEGIN LOOKING
 0BC4'    00            C              DB      00H             ;FOR @
                        C              CHAR?                   ;GET A CHARACTER
 0BC5'    D4            C+
 0BC6'    0145'         C+
 0BC8'    9C            C+
 0BC9'    CA 0939'      C+
 0BCC'    8C            C+
 0BCD'    FB 40         C              XRI     "@"             ;IS IT AN "@" ?
 0BCF'    CA 0BC2'      C              LBNZ    AT?             ;IF NOT KEEP LOOKING
 0BD2'    C0 03B0'      C              LBR     SAIL            ;IF SO, DE-ADDRESS
                        C       ;
                                ;
                        C              INCLUDE RAMTST.MAC
                        C       ;              ***************
                        C       ;              * RAMTST.MAC *
                        C       ;              ***************
                        C       ;
                        C       ;
                        C       ;          ___________________________
                        C       ;          + TEST RAM OVER SPECIFIED AREA +
                        C       ;          ___________________________
                        C       ;
                        C       ;The response to an "R" command is to first test the system
                        C       ;flag, then, if this flag is set, to prompt for a start address
                        C       ;and block size. If the system flag was not set, exit and
                        C       ;indicate an operator error since the memory was protected.
                        C       ;After the start address and block size have been input, type
                        C       ;"OK ? (Y/N)". If the operator types a "Y" in response to this
                        C       ;question proceed with the RAM TEST. Load the entire specified
                        C       ;block of RAM with a random number and verify a byte at a time.
```

```
                         C      ;Type the address of each compare failure and its XOR data.
                         C      ;Repeat the tests changing the random number with each pass.
                         C      ;Program will exit upon detecting a UART error, but since
                         C      ;interrupts have been disabled, the system MUST be reset.
                         C      ;
 0BD5'                   C      RAMTST: TYPMSG  RMTST           ;TYPE "am test"
 0BD5'   D4              C+
 0BD6'   00CB'           C+
 0BD8'   0596'           C+
 0BDA'   9C              C+
 0BDB'   CA 0939'        C+
                         C              GETFLG                  ;IS THE SYSTEM LOCKED?
 0BDE'   F8 01           C+
 0BE0'   A7              C+
 0BE1'   07              C+
 0BE2'   F6              C              SHR                     ;IF SO, TYPE THE ERROR
 0BE3'   C3 0BE9'        C              LBDF    RSPEC           ;MESSAGE AND PROMPT
 0BE6'   C0 08C6'        C              LBR     NORUN           ;OTHERWISE, RESET THE
 0BE9'   E7              C      RSPEC:  SEX     R7              ;SYSTEM LOCK FLAG
 0BEA'   F8 FE           C              LDI     0FEH
 0BEC'   F2              C              AND
 0BED'   57              C              STR     R7              ;THEN PROMPT FOR
                         C              CALL    GET2HX          ;START ADDRESS AND BLOCK SIZE
 0BEE'   D4              C+
 0BEF'   023C'           C+
                         C              ERROR?                  ;REACT TO ERRORS
 0BF1'   9C              C+
 0BF2'   CA 0939'        C+
 0BF5'   9A              C              GHI     RA              ;SAVE START ADDRESS
 0BF6'   B8              C              PHI     R8              ;USING REGISTER R8
 0BF7'   8A              C              GLO     RA
 0BF8'   A8              C              PLO     R8
 0BF9'   9B              C              GHI     RB              ;SAVE BLOCK SIZE
 0BFA'   B9              C              PHI     R9              ;USING REGISTER R9
 0BFB'   8B              C              GLO     RB
 0BFC'   A9              C              PLO     R9
                         C              CALL    ASKOK           ;ASK FINAL PERMISSION
 0BFD'   D4              C+
 0BFE'   0113'           C+
                         C              ERROR?                  ;REACT TO UART ERRORS
 0C00'   9C              C+
 0C01'   CA 0939'        C+
 0C04'   8C              C              GLO     RC              ;GET ANSWER
 0C05'   CA 08ED'        C              LBNZ    PRMOUT          ;EXIT IF NOT YES
                         C              TYPMSG  CRLF            ;OTHERWISE, TYPE A CR/LF
 0C08'   D4              C+
 0C09'   00CB'           C+
 0C0B'   03D0'           C+
 0C0D'   9C              C+
 0C0E'   CA 0939'        C+
 0C11'   E3              C              SEX     R3              ;DISABLE INTERRUPTS
 0C12'   71              C              DIS                     ;TO STOP THE CLOCK
 0C13'   33              C              DB      33H             ;AND FALSE RAM ERRORS
 0C14'   F8 73           C              LDI     73H             ;AND SET RANDOM KEY =
 0C16'   BF              C              PHI     RF              ; 01110011
 0C17'   F8 0C'          C              LDI     HIGH    (LDREGS)
```

```
0C19'   B0          C           PHI     R0                  ;SET UP R0 TO BE A
0C1A'   F8 27'      C           LDI     LOW     (LDREGS)
0C1C'   A0          C           PLO     R0                  ;SUBROUTINE POINTER
0C1D'   F8 0C'      C           LDI     HIGH    (RAND)
0C1F'   BE          C           PHI     RE                  ;SET UP RE TO BE A
0C20'   F8 35'      C           LDI     LOW     (RAND)
0C22'   AE          C           PLO     RE                  ;SUBROUTINE POINTER
0C23'   C0 0C5C'    C           LBR     NCYCLE              ;EXECUTE RAM TEST
                    C   ;
                    C   ;This is a subroutine which will load the next  random
                    C   ;key to the high half of register RD, the start address
                    C   ;to register RA, and the block size to register RB.
                    C   ;Using a subroutine here slows execution, but saves PROM.
                    C   ;
0C26'   D3          C   LTOP:   SEP     R3                  ;BACK TO RAM TEST
0C27'   9F          C   LDREGS: GHI     RF                  ;GET NEW KEY
0C28'   BD          C           PHI     RD                  ;PASS TO RD
0C29'   98          C           GHI     R8                  ;GET START ADDRESS
0C2A'   BA          C           PHI     RA
0C2B'   88          C           GLO     R8                  ;PASS TO RA
0C2C'   AA          C           PLO     RA
0C2D'   99          C           GHI     R9                  ;GET BLOCK SIZE
0C2E'   BB          C           PHI     RB
0C2F'   89          C           GLO     R9                  ;PASS TO RB
0C30'   AB          C           PLO     RB
0C31'   C0 0C26'    C           LBR     LTOP                ;RETURN
                    C   ;
                    C   ;This is a subroutine which will return with a random
                    C   ;number in the high half of register RD. The random
                    C   ;number is generated by right shifting the modulo 2 sum
                    C   ;of bits 0,2,3,and 4 to bit 7.
                    C   ;
0C34'   D3          C   RTOP:   SEP     R3                  ;BACK TO RAM TEST
0C35'   F8 00       C   RAND:   LDI     00H                 ;KEY IS IN RD HIGH
0C37'   AD          C           PLO     RD
0C38'   9D          C           GHI     RD                  ;SET TO FFH IF KEY
0C39'   CA 0C3F'    C           LBNZ    ONNO0               ;IS NOW EQUAL TO 00
0C3C'   F8 FF       C           LDI     0FFH
0C3E'   BD          C           PHI     RD
0C3F'   F6          C   ONNO0:  SHR                         ;TEST BIT 1
0C40'   CB 0C44'    C           LBNF    ONNO1               ;ADD ONE IF SET
0C43'   1D          C           INC     RD
0C44'   F6          C   ONNO1:  SHR                         ;SKIP BIT 1
0C45'   F6          C           SHR                         ;TEST BIT 2
0C46'   CB 0C4A'    C           LBNF    ONNO2               ;ADD ONE IF SET
0C49'   1D          C           INC     RD
0C4A'   F6          C   ONNO2:  SHR                         ;TEST BIT 3
0C4B'   CB 0C4F'    C           LBNF    ONNO3               ;ADD ONE IF SET
0C4E'   1D          C           INC     RD
0C4F'   F6          C   ONNO3:  SHR                         ;TEST BIT 4
0C50'   CB 0C54'    C           LBNF    ONNO4               ;ADD ONE IF SET
0C53'   1D          C           INC     RD
0C54'   8D          C   ONNO4:  GLO     RD                  ;GET RESULT OF SUM
0C55'   F6          C           SHR
0C56'   9D          C           GHI     RD                  ;SHIFT IT TO RD HIGH
0C57'   76          C           RSHR
```

```
0C58'   BD          C           PHI     RD
0C59'   C0 0C34'    C           LBR     RTOP           ;RETURN
                    C   ;
0C5C'   D0          C   NCYCLE: SEP     R0             ;LOAD REGISTERS
0C5D'   DE          C   WRITE:  SEP     RE             ;GENERATE A RANDOM NUMBER
0C5E'   9D          C           GHI     RD             ;AND
0C5F'   5A          C           STR     RA             ;STORE IT
0C60'   1A          C           INC     RA             ;MOVE POINTER
0C61'   2B          C           DEC     RB             ;COUNT OPERATION
0C62'   9B          C           GHI     RB             ;CONTINUE UNTIL
0C63'   CA 0C5D'    C           LBNZ    WRITE          ;SPECIFIED BLOCK
0C66'   8B          C           GLO     RB             ;IS LOADED WITH
0C67'   CA 0C5D'    C           LBNZ    WRITE          ;RANDOM NUMBERS
0C6A'   D0          C   VERIFY: SEP     R0             ;RESET ALL REGISTERS
0C6B'   DE          C   VERCYC: SEP     RE             ;GENERATE A RANDOM NUMBER
0C6C'   9D          C           GHI     RD             ;COMPARE IT WITH RAM DATA
0C6D'   EA          C           SEX     RA             ;IF DIFFERENT THERE IS
0C6E'   F3          C           XOR                    ;AN ERROR
0C6F'   CA 0C8D'    C           LBNZ    WRTERR
0C72'   1A          C   NXTLOC: INC     RA             ;MOVE POINTER
0C73'   2B          C           DEC     RB             ;COUNT OPERATION
0C74'   9B          C           GHI     RB             ;CONTINUE UNTIL
0C75'   CA 0C6B'    C           LBNZ    VERCYC         ;ALL SPECIFIED RAM
0C78'   8B          C           GLO     RB             ;HAS BEEN EXAMINED
0C79'   CA 0C6B'    C           LBNZ    VERCYC
                    C           TYPMSG  ASTK           ;TYPE AN * AT END OF PASS
0C7C'   D4          C+
0C7D'   00CB'       C+
0C7F'   0594'       C+
0C81'   9C          C+
0C82'   CA 0939'    C+
0C85'   9F          C           GHI     RF             ;GET LAST KEY
0C86'   BD          C           PHI     RD             ;RANDOMIZE IT
0C87'   DE          C           SEP     RE             ;THE RESULT
0C88'   9D          C           GHI     RD             ;BECOMES NEW KEY
0C89'   BF          C           PHI     RF
0C8A'   C0 0C5C'    C           LBR     NCYCLE         ;MAKE ANOTHER PASS
0C8D'   AD          C   WRTERR: PLO     RD             ;SAVE RESULT OF XOR
                    C           TYPMSG  CRLFSP         ;MOVE TO NEXT LINE
0C8E'   D4          C+
0C8F'   00CB'       C+
0C91'   03D6'       C+
0C93'   9C          C+
0C94'   CA 0939'    C+
0C97'   9A          C           GHI     RA             ;TYPE CURRENT ADDRESS
0C98'   AC          C           PLO     RC
                    C           CALL    TYPEC          ;HIGH BYTE
0C99'   D4          C+
0C9A'   021F'       C+
                    C           ERROR?                 ;REACT TO UART ERRORS
0C9C'   9C          C+
0C9D'   CA 0939'    C+
0CA0'   8A          C           GLO     RA
0CA1'   AC          C           PLO     RC
                    C           CALL    TYPEC          ;AND LOW BYTE
0CA2'   D4          C+
0CA3'   021F'       C+
                    C           ERROR?                  ;REACT TO ERRORS
0CA5'   9C          C+
0CA6'   CA 0939'    C+
                    C           TYPMSG  SPSP            ;TYPE TWO SPACES AND
0CA9'   D4          C+
0CAA'   00CB'       C+
0CAC'   03D3'       C+
0CAE'   9C          C+
0CAF'   CA 0939'    C+
0CB2'   8D          C           GLO     RD              ;THE RESULT OF THE XOR
0CB3'   AC          C           PLO     RC
                    C           CALL    TYPEC
0CB4'   D4          C+
0CB5'   021F'       C+
                    C           ERROR?                  ;REACT TO UART ERRORS
0CB7'   9C          C+
0CB8'   CA 0939'    C+
0CBB'   C0 0C72'    C           LBR     NXTLOC          ;TEST THE NEXT LOCATION
                    C     ;
                          ;
                    C           INCLUDE ISCEDUL.MAC
                    C     ;
                    C     ;             **************
                    C     ;             * SCEDUL.MAC *
                    C     ;             **************
                    C     ;
                    C     ; ______________________________
                    C     ; + SET THE INTERROGATOR SCHEDULE +
                    C     ; ______________________________
                    C     ;
                    C     ;This block of code will set the operating schedule
                    C     ;of the interrogator. Two parameters are set via prompts,
                    C     ;the start time, and the measurement interval.
                    C     ;
                    C     ;Define decimal and ASCII start times in RAM
                    C     ;
FF30                C     DSHD    EQU     GLOBAL+30H
FF36                C     DSUM    EQU     DSHD+06H
FF38                C     ASHD    EQU     GLOBAL+38H
FF3C                C     ASTH    EQU     ASHD+04H
FF3F                C     ASTM    EQU     ASTH+03H
FF3E                C     ASUM    EQU     ASHD+06H
                    C     ;
                    C     ;Define decimal and ASCII measurement interval in RAM
                    C     ;
FF45                C     DIHM    EQU     GLOBAL+45H
FF4A                C     AIHM    EQU     DIHM+05H
                    C     ;
                    C     ;Define a location in ram to hold the hex equivalent
                    C     ;of the measurement interval.
                    C     ;
FF24                C     HEXMI   EQU     GLOBAL+24H
                    C     ;
                    C     ;Define another location for the number of minutes
                    C     ;in hex to the next measurement. This number is only
```

```
                          C      ;valid if the schedule is active.
                          C      ;
 FF26                     C      MINOW    EQU     HEXMI+2
                          C      ;
 0CBE'    F8 36           C      LDSCED:  LDI     LOW      (DSUM)   ;POINT AT START TIME
 0CC0'    A7              C               PLO     R7                ;AND LOAD ZEROS
 0CC1'    E7              C               SEX     R7
 0CC2'    F8 00           C               LDI     00H
 0CC4'    73              C               STXD
 0CC5'    73              C               STXD
 0CC6'    73              C               STXD
 0CC7'    73              C               STXD
 0CC8'    73              C               STXD
 0CC9'    73              C               STXD
 0CCA'    57              C               STR     R7                ;R7 WAS POINTING TO SDHD
 0CCB'    F8 43           C               LDI     LOW      (GOFLG)  ;DEARM SCHEDULER
 0CCD'    A7              C               PLO     R7                ;BY LOADING ZEROS
 0CCE'    F8 00           C               LDI     00H               ;TO BOTH HALVES OF
 0CD0'    57              C               STR     R7                ;OF THE GO FLAG
 0CD1'    17              C               INC     R7
 0CD2'    57              C               STR     R7
                          C               TYPMSG  STDAY             ;PROMPT FOR START DAY
 0CD3'    D4              C+
 0CD4'    00CB'           C+
 0CD6'    0465'           C+
 0CD8'    9C              C+
 0CD9'    CA 0939'        C+
                          C               CALL    INDEC             ;GET START DAY AND
 0CDC'    D4              C+
 0CDD'    0196'           C+
 0CDF'    FF32            C               DW      DSHD+2            ;STORE
 0CE1'    03              C               DB      03H               ;(THREE DIGITS)
 0CE2'    9C              C               GHI     RC                ;LOOK FOR ERRORS
 0CE3'    FA FE           C               ANI     0FEH              ;MASK NON-DECIMAL BIT
 0CE5'    CA 0939'        C               LBNZ    ERVEC             ;EXIT ON ERROR
 0CE8'    8C              C               GLO     RC                ;TEST FOR SPACE
 0CE9'    FB 20           C               XRI     SPACE             ;CONTINUE IF FOUND
 0CEB'    CA 08F9'        C               LBNZ    ERROUT            ;OTHERWISE,INDICATE ERROR
                          C               TYPMSG  STHOUR            ;PROMPT FOR START HOUR
 0CEE'    D4              C+
 0CEF'    00CB'           C+
 0CF1'    0478'           C+
 0CF3'    9C              C+
 0CF4'    CA 0939'        C+
                          C               CALL    INDEC             ;GET START HOUR AND
 0CF7'    D4              C+
 0CF8'    0196'           C+
 0CFA'    FF34            C               DW      DSHD+4            ;STORE
 0CFC'    02              C               DB      02H               ;(TWO DIGITS)
 0CFD'    9C              C               GHI     RC                ;LOOK FOR UART ERRORS
 0CFE'    FA FE           C               ANI     0FEH              ;BY MASKING NON-DECIMAL
 0D00'    CA 0939'        C               LBNZ    ERVEC             ;EXIT IF FOUND
 0D03'    8C              C               GLO     RC                ;LOOK FOR A SPACE
 0D04'    FB 20           C               XRI     SPACE             ;CONTINUE IF FOUND
 0D06'    CA 08F9'        C               LBNZ    ERROUT            ;OTHERWISE INDICATE ERROR
                          C               TYPMSG  STMIN             ;PROMPT FOR START MINUTE
```

```
0D09'   D4            C+
0D0A'   00CB'         C+
0D0C'   0482'         C+
0D0E'   9C            C+
0D0F'   CA 0939'      C+
                      C                CALL    INDEC           ;GET START MINUTE AND
0D12'   D4            C+
0D13'   0196'         C+
0D15'   FF36          C                DW      DSHD+6          ;STORE
0D17'   02            C                DB      02H             ;TWO DIGITS
0D18'   9C            C                GHI     RC              ;LOOK FOR UART ERRORS
0D19'   FA FE         C                ANI     0FEH            ;MASK NON-DECIMAL FLAG
0D1B'   CA 0939'      C                LBNZ    ERVEC           ;EXIT ON ERROR
0D1E'   8C            C                GLO     RC              ;LOOK FOR
0D1F'   FB 20         C                XRI     SPACE           ;A SPACE
0D21'   C2 0D2A'      C                LBZ     INMINT          ;IF FOUND CONTINUE
0D24'   8C            C                GLO     RC              ;OTHERWISE LOOK FOR A
0D25'   FB 0D         C                XRI     CR              ;CARRIAGE RETURN
0D27'   CA 08F9'      C                LBNZ    ERROUT          ;ERROR IF NOT FOUND
                      C        ;
                      C        ;Start date and time are now in RAM. Measurement
                      C        ;interval is next prompted for, input, converted
                      C        ;to hex, and stored in two locations.
                      C        ;
0D2A'                 C        INMINT: TYPMSG  MEAINT          ;PROMPT FOR MEAS. INT.
0D2A'   D4            C+
0D2B'   00CB'         C+
0D2D'   048E'         C+
0D2F'   9C            C+
0D30'   CA 0939'      C+
                      C                CALL    INDEC           ;GET MEASUREMENT INTERVAL
0D33'   D4            C+
0D34'   0196'         C+
0D36'   FF47          C                DW      DIHM+2          ;AND STORE
0D38'   03            C                DB      03H             ;(THREE DIGITS)
0D39'   9C            C                GHI     RC              ;LOOK FOR UART ERRORS
0D3A'   FA FE         C                ANI     0FEH            ;AND EXIT IF FOUND
0D3C'   CA 0939'      C                LBNZ    ERVEC           ;INDICATE AN ERROR
0D3F'   8C            C                GLO     RC              ;LOOK FOR A SPACE
0D40'   FB 20         C                XRI     SPACE           ;CONTINUE IF FOUND
0D42'   C2 0D4E'      C                LBZ     BCDHEX          ;OTHERWISE,
0D45'   8C            C                GLO     RC              ;LOOK FOR A
0D46'   FB 0D         C                XRI     CR              ;CARRIAGE RETURN
0D48'   C2 0D4E'      C                LBZ     BCDHEX          ;CONT. IF FOUND, OTHERWISE
0D4B'   C0 08F9'      C                LBR     ERROUT          ;INDICATE AN OPERATOR ERROR
0D4E'   E7            C        BCDHEX: SEX     R7              ;POINT AT DEC. INT. U.M.
0D4F'   F8 47         C                LDI     LOW     (DIHM+2)
0D51'   A7            C                PLO     R7              ;CONVERT MEASUREMENT
0D52'   F8 00         C                LDI     00H             ;INTERVAL TO HEX
0D54'   AA            C                PLO     RA
0D55'   BA            C                PHI     RA              ;ZERO REGISTER RA
0D56'   07            C                LDN     R7              ;GET UNITS DIGIT
0D57'   AA            C                PLO     RA
0D58'   27            C                DEC     R7              ;POINT AT TENS DIGIT
0D59'   07            C                LDN     R7              ;SET ADD COUNTER
0D5A'   AC            C                PLO     RC              ;AND TEST FOR ZERO
```

```
0D5B'   C2 0D67'     C            LBZ     AD100           ;ADVANCE IF ZERO
0D5E'   8A           C    AD10:   GLO     RA              ;OTHERWISE, ADD 0AH
0D5F'   FC 0A        C            ADI     0AH             ;TO ACCUMULATOR
0D61'   AA           C            PLO     RA
0D62'   2C           C            DEC     RC              ;COUNT OPERATION
0D63'   8C           C            GLO     RC
0D64'   CA 0D5E'     C            LBNZ    AD10            ;CONTINUE TILL ZERO
0D67'   27           C    AD100:  DEC     R7              ;POINT AT HUNDREDS DIGIT
0D68'   07           C            LDN     R7              ;SET ADD COUNTER
0D69'   AC           C            PLO     RC
0D6A'   C2 0D7A'     C            LBZ     BHDONE          ;IF ZERO CONVERT IS DONE
0D6D'   8A           C    NXTADD: GLO     RA              ;OTHERWISE, ADD 64H
0D6E'   FC 64        C            ADI     64H             ;TO ACCUMULATOR
0D70'   AA           C            PLO     RA
0D71'   9A           C            GHI     RA
0D72'   7C 00        C            ADCI    00H             ;INCLUDE CARRY BIT
0D74'   BA           C            PHI     RA
0D75'   2C           C            DEC     RC              ;COUNT OPERATION
0D76'   8C           C            GLO     RC
0D77'   CA 0D6D'     C            LBNZ    NXTADD          ;CONTINUE TILL ZERO
0D7A'   F8 27        C    BHDONE: LDI     LOW     (MINOW+1)
0D7C'   A7           C            PLO     R7              ;STORE RESULT OF
0D7D'   8A           C            GLO     RA              ;CONVERT AT MINOW
0D7E'   73           C            STXD
0D7F'   9A           C            GHI     RA
0D80'   73           C            STXD
0D81'   8A           C            GLO     RA              ;AND AT HEXMI
0D82'   73           C            STXD
0D83'   9A           C            GHI     RA
0D84'   57           C            STR     R7
0D85'   8A           C            GLO     RA              ;TEST RESULT AND IF
0D86'   FF 03        C            SMI     03H             ;NOT GREATER THAN
0D88'   9A           C            GHI     RA              ;2 TYPE AN ERROR
0D89'   7F 00        C            SMBI    00H             ;MESSAGE, OTHERWISE,
0D8B'   C3 0D9A'     C            LBDF    SETAD           ;SET ADDRESS POINTER
                     C            TYPMSG  MIMIN
0D8E'   D4           C+
0D8F'   00CB'        C+
0D91'   0535'        C+
0D93'   9C           C+
0D94'   CA 0939'     C+
0D97'   C0 0D2A'     C            LBR     INMINT
                     C    ;
                     C    ;Set the data address pointer to its first location
                     C    ;
0D9A'   F8 0F        C    SETAD:  LDI     LOW     (STRADD+1)
0D9C'   A7           C            PLO     R7
0D9D'   F8 00        C            LDI     00H             ;STORE ADDRESS OF
0D9F'   73           C            STXD                    ;FIRST DATA BYTE
0DA0'   F8 10        C            LDI     10H
0DA2'   57           C            STR     R7
                     C            TYPMSG  SPSP            ;ASK IF THIS SCHEDULE IS OK
0DA3'   D4           C+
0DA4'   00CB'        C+
0DA6'   03D3'        C+
0DA8'   9C           C+
ODA9'   CA 0939'      C+
                      C             TYPMSG  OK?
ODAC'   D4            C+
ODAD'   00CB'         C+
ODAF'   0423'         C+
ODB1'   9C            C+
ODB2'   CA 0939'      C+
                      C             CHAR?                     ;GET RESPONSE
ODB5'   D4            C+
ODB6'   0145'         C+
ODB8'   9C            C+
ODB9'   CA 0939'      C+
ODBC'   8C            C+
ODBD'   FB 59         C             XRI     "Y"               ;IS IT YES ?
ODBF'   CA 0DCB'      C             LBNZ    DEARM             ;IF NOT EXIT
ODC2'   F8 43         C             LDI     LOW     (GOFLG)   ;OTHERWISE, ARM
ODC4'   A7            C             PLO     R7                ;THE SCHEDULER
ODC5'   F8 AA         C             LDI     0AAH              ;BY SETTING HI HALF
ODC7'   57            C             STR     R7                ;GO FLAG THEN
ODC8'   C0 08ED'      C             LBR     PRMOUT            ;GET NEXT COMMAND
ODCB'   F8 43         C    DEARM:   LDI     LOW     (GOFLG)   ;ANSWER WAS NOT
ODCD'   A7            C             PLO     R7                ;YES, SO RESET
ODCE'   F8 00         C             LDI     00H               ;GO FLAG AND EXIT
ODD0'   57            C             STR     R7
ODD1'   17            C             INC     R7
ODD2'   57            C             STR     R7
ODD3'   C0 08ED'      C             LBR     PRMOUT
                      C    ;
                      C    ;The response to a ?S command is to convert the
                      C    ;start time and transmission interval to an ASCII
                      C    ;message string, type the message, and return to CMND.
                      C    ;
ODD6'   F8 30         C    QRYSCED:LDI      LOW     (DSHD)    ;CONVERT START TIME
ODD8'   A7            C             PLO     R7                ;TO ASCII
ODD9'   F8 38         C             LDI     LOW     (ASHD)    ;STORE AT ASHD
ODDB'   AA            C             PLO     RA                ;USING RA AS A POINTER
ODDC'   97            C             GHI     R7
ODDD'   BA            C             PHI     RA
                      C             CALL    DTOA              ;CONVERT DAYS
ODDE'   D4            C+
ODDF'   0095'         C+
ODE1'   03            C             DB      03H
ODE2'   F8 7E         C             LDI     STOP              ;STORE A STOP
ODE4'   5A            C             STR     RA                ;BETWEEN DAYS AND
ODE5'   1A            C             INC     RA                ;HOURS
                      C             CALL    DTOA              ;CONVERT HOURS
ODE6'   D4            C+
ODE7'   0095'         C+
ODE9'   02            C             DB      02H
ODEA'   F8 7E         C             LDI     STOP              ;STORE A STOP
ODEC'   5A            C             STR     RA                ;BETWEEN HOURS
ODED'   1A            C             INC     RA                ;AND MINUTES
                      C             CALL    DTOA              ;CONVERT MINUTES
ODEE'   D4            C+
ODEF'   0095'         C+
ODF1'   02            C             DB      02H
```

```
  0DF2'     F8 7E             C                LDI     STOP             ;STORE A STOP
  0DF4'     5A                C                STR     RA               ;BETWEEN MINUTES
  0DF5'     1A                C                INC     RA               ; AND TRAN. INT.
  0DF6'     F8 45             C                LDI     LOW     (DIHM)   ;CONVERT TANS. INT.
  0DF8'     A7                C                PLO     R7               ;AND STORE AT AIHM
  0DF9'     F8 4A             C                LDI     LOW     (AIHM)   ;USING RA AS A POINTER
  0DFB'     AA                C                PLO     RA
                              C                CALL    DTOA             ;CONVERT TRANSMISSION
  0DFC'     D4                C+
  0DFD'     0095'             C+
  0DFF'     03                C                DB      03H              ;INTERVAL TO ASCII
  0E00'     F8 7E             C                LDI     STOP             ;STORE MESSAGE
  0E02'     5A                C                STR     RA               ;TERMINATION CHARACTER.
                              C     ;
                              C     ;Type current time.
                              C     ;
                              C                TYPMSG  SPSP             ;TYPE TWO SPACES
  0E03'     D4                C+
  0E04'     00CB'             C+
  0E06'     03D3'             C+
  0E08'     9C                C+
  0E09'     CA 0939'          C+
  0E0C'     F8 18             C                LDI     LOW     (TICK+1)
  0E0E'     AA                C                PLO     RA               ;CONVERT TIME TO ASCII
  0E0F'     97                C                GHI     R7               ;USING RA AS A POINTER
  0E10'     BA                C                PHI     RA
  0E11'     F8 10             C                LDI     LOW     (HD)
  0E13'     A7                C                PLO     R7
                              C                CALL    DTOA             ;CONVERT DAYS
  0E14'     D4                C+
  0E15'     0095'             C+
  0E17'     03                C                DB      03H
  0E18'     F8 20             C                LDI     SPACE
  0E1A'     5A                C                STR     RA
  0E1B'     1A                C                INC     RA
                              C                CALL    DTOA             ;CONVERT HOURS
  0E1C'     D4                C+
  0E1D'     0095'             C+
  0E1F'     02                C                DB      02H
  0E20'     F8 3A             C                LDI     ":"
  0E22'     5A                C                STR     RA
  0E23'     1A                C                INC     RA
                              C                CALL    DTOA             ;CONVERT MINUTES
  0E24'     D4                C+
  0E25'     0095'             C+
  0E27'     02                C                DB      02H
  0E28'     F8 7E             C                LDI     STOP
  0E2A'     5A                C                STR     RA
                              C                TYPMSG  SAT              ;SAY AT
  0E2B'     D4                C+
  0E2C'     00CB'             C+
  0E2E'     058C'             C+
  0E30'     9C                C+
  0E31'     CA 0939'          C+
                              C                TYPMSG  NXTM             ;TYPE TIME
  0E34'     D4                C+
 0E35'   00CB'          C+
 0E37'   FF18           C+
 0E39'   9C             C+
 0E3A'   CA 0939'       C+
                        C            TYPMSG  STDAY           ;TYPE CURRENT SCHEDULE
 0E3D'   D4             C+
 0E3E'   00CB'          C+
 0E40'   0465'          C+
 0E42'   9C             C+
 0E43'   CA 0939'       C+
                        C            TYPMSG  ASHD
 0E46'   D4             C+
 0E47'   00CB'          C+
 0E49'   FF38           C+
 0E4B'   9C             C+
 0E4C'   CA 0939'       C+
                        C            TYPMSG  STHOUR
 0E4F'   D4             C+
 0E50'   00CB'          C+
 0E52'   0478'          C+
 0E54'   9C             C+
 0E55'   CA 0939'       C+
                        C            TYPMSG  ASTH
 0E58'   D4             C+
 0E59'   00CB'          C+
 0E5B'   FF3C           C+
 0E5D'   9C             C+
 0E5E'   CA 0939'       C+
                        C            TYPMSG  STMIN
 0E61'   D4             C+
 0E62'   00CB'          C+
 0E64'   0482'          C+
 0E66'   9C             C+
 0E67'   CA 0939'       C+
                        C            TYPMSG  ASTM
 0E6A'   D4             C+
 0E6B'   00CB'          C+
 0E6D'   FF3F           C+
 0E6F'   9C             C+
 0E70'   CA 0939'       C+
                        C            TYPMSG  MEAINT
 0E73'   D4             C+
 0E74'   00CB'          C+
 0E76'   048E'          C+
 0E78'   9C             C+
 0E79'   CA 0939'       C+
                        C            TYPMSG  AIHM
 0E7C'   D4             C+
 0E7D'   00CB'          C+
 0E7F'   FF4A           C+
 0E81'   9C             C+
 0E82'   CA 0939'       C+
                        C            TYPMSG  SCDMSG
 0E85'   D4             C+
 0E86'   00CB'          C+
 0E88'   04B3'          C+
0E8A'   9C              C+
0E8B'   CA 0939'        C+
0E8E'   F8 43           C                LDI     LOW     (GOFLG)
0E90'   A7              C                PLO     R7                      ;IF GO FLAG IS SET
0E91'   47              C                LDA     R7                      ;SAY ACTIVE, AND
0E92'   FB AA           C                XRI     0AAH                    ;INDICATE THE NUMBER
0E94'   CA 0EF9'        C                LBNZ    SNARM                   ;OF MINUTES TO THE NEXT
0E97'   07              C                LDN     R7                      ;MEASUREMENT, OTHERWISE
0E98'   FB AA           C                XRI     0AAH                    ;TYPE THE CURRENT
0E9A'   CA 0F05'        C                LBNZ    SARMI                   ;SYSTEM STATUS AND EXIT
                        C                TYPMSG  ACTIVE
0E9D'   D4              C+
0E9E'   00CB'           C+
0EA0'   04C4'           C+
0EA2'   9C              C+
0EA3'   CA 0939'        C+
0EA6'   F8 26           C                LDI     LOW     (MINOW)
0EA8'   A7              C                PLO     R7                      ;GET HEX MINOW
0EA9'   47              C                LDA     R7
0EAA'   BA              C                PHI     RA                      ;AND PLACE IN RA
0EAB'   07              C                LDN     R7
0EAC'   AA              C                PLO     RA
0EAD'   9A              C                GHI     RA
0EAE'   AC              C                PLO     RC
                        C                CALL    TYPEC                   ;TYPE HI BYTE
0EAF'   D4              C+
0EB0'   021F'           C+
                        C                ERROR?                          ;REACT TO ERRORS
0EB2'   9C              C+
0EB3'   CA 0939'        C+
0EB6'   8A              C                GLO     RA                      ;GET HEX MINOW LO
0EB7'   AC              C                PLO     RC
                        C                CALL    TYPEC                   ;TYPE LO BYTE
0EB8'   D4              C+
0EB9'   021F'           C+
                        C                ERROR?                          ;REACT TO ERRORS
0EBB'   9C              C+
0EBC'   CA 0939'        C+
                        C                TYPMSG  MINREM
0EBF'   D4              C+
0EC0'   00CB'           C+
0EC2'   04D1'           C+
0EC4'   9C              C+
0EC5'   CA 0939'        C+
                        C                TYPMSG  PNTR                    ;INDICATE POINTER LOCATION
0EC8'   D4              C+
0EC9'   00CB'           C+
0ECB'   0504'           C+
0ECD'   9C              C+
0ECE'   CA 0939'        C+
0ED1'   F8 0E           C                LDI     LOW     (STRADD)
0ED3'   A7              C                PLO     R7
0ED4'   47              C                LDA     R7                      ;GET CURRENT STORE ADDRESS
0ED5'   BA              C                PHI     RA
0ED6'   07              C                LDN     R7
0ED7'   AA              C                PLO     RA
```

```
0ED8'   9A          C               GHI     RA
0ED9'   AC          C               PLO     RC
                    C               CALL    TYPEC               ;TYPE HI BYTE
0EDA'   D4          C+
0EDB'   021F'       C+
                    C               ERROR?                      ;REACT TO ERRORS
0EDD'   9C          C+
0EDE'   CA 0939'    C+
0EE1'   8A          C               GLO     RA
0EE2'   AC          C               PLO     RC
                    C               CALL    TYPEC               ;TYPE LO BYTE
0EE3'   D4          C+
0EE4'   021F'       C+
                    C               ERROR?                      ;REACT TO ERRORS
0EE6'   9C          C+
0EE7'   CA 0939'    C+
0EEA'   C0 08ED'    C               LBR     PRMOUT              ;GET NEXT COMMAND
0EED'               C       SAYIDL: TYPMSG  IDLE1               ;GO FLAG NOT SET
0EED'   D4          C+
0EEE'   00CB'       C+
0EF0'   04FE'       C+
0EF2'   9C          C+
0EF3'   CA 0939'    C+
0EF6'   C0 08ED'    C               LBR     PRMOUT              ;SAY IDLE AND EXIT
0EF9'               C       SNARM:  TYPMSG  NOTARM              ;SAY NOT ARMED
0EF9'   D4          C+
0EFA'   00CB'       C+
0EFC'   0582'       C+
0EFE'   9C          C+
0EFF'   CA 0939'    C+
0F02'   C0 08ED'    C               LBR     PRMOUT              ;AND EXIT
0F05'               C       SARMI:  TYPMSG  ARMIDL              ;SAY ARMED BUT IDLE
0F05'   D4          C+
0F06'   00CB'       C+
0F08'   056D'       C+
0F0A'   9C          C+
0F0B'   CA 0939'    C+
0F0E'   C0 08ED'    C               LBR     PRMOUT              ;AND EXIT
                    C       ;
                    C       ;The response to a "!IDLE" command is to test the go
                    C       ;flag and if set, reset it. If the go flag is already
                    C       ;reset, the message "Scheduler was NOT active !!" will be
                    C       ;sent.
                    C       ;
0F11'   F8 43       C       GFTOO:  LDI     LOW     (GOFLG)
0F13'   A7          C               PLO     R7                  ;GET GO FLAG
0F14'   07          C               LDN     R7                  ;IS IT SET ?
0F15'   FB AA       C               XRI     0AAH                ;IF SO RESET IT
0F17'   CA 0F2B'    C               LBNZ    SAYNOT              ;OTHERWISE TYPE
0F1A'   F8 00       C               LDI     00H                 ;NOT ACTIVE MESSAGE
0F1C'   57          C               STR     R7                  ;RESET HI AND LO
0F1D'   17          C               INC     R7                  ;OF GO FLAG
0F1E'   57          C               STR     R7                  ;THEN
                    C               TYPMSG  OK                  ;SAY OK AND GET
0F1F'   D4          C+
0F20'   00CB'       C+
0F22'   041F'      C+
0F24'   9C         C+
0F25'   CA 0939'   C+
0F28'   C0 08ED'   C            LBR     PRMOUT          ;NEXT COMMAND
0F2B'              C    SAYNOT: TYPMSG  NOTACT          ;FLAG WAS NOT SET
0F2B'   D4         C+
0F2C'   00CB'      C+
0F2E'   0516'      C+
0F30'   9C         C+
0F31'   CA 0939'   C+
0F34'   C0 08ED'   C            LBR     PRMOUT          ;GET NEXT COMMAND
                   C    ;
                   C    ;The response to a "!PING" command is to first ask "OK ?"
                   C    ;and if a "Y" is the answer to trigger the pinger. Any other
                   C    ;answer will cause an exit to CMND.
                   C    ;
0F37'              C    TXMIT:  CALL    ASKOK           ;ASK PERMISSION
0F37'   D4         C+
0F38'   0113'      C+
                   C            ERROR?                  ;REACT TO UART ERRORS
0F3A'   9C         C+
0F3B'   CA 0939'   C+
0F3E'   8C         C            GLO     RC              ;IS IT YES ?
0F3F'   CA 08ED'   C            LBNZ    PRMOUT          ;IF NOT EXIT
0F42'   E3         C            SEX     R3
0F43'   63         C            OUT     PING            ;SEND PING
0F44'   00         C            DB      00H
0F45'   C0 08ED'   C            LBR     PRMOUT          ;GET NEXT COMMAND
                   C    ;
                        ;
                   C            INCLUDE IMAIN.MAC
                   C    ;       ************
                   C    ;       * MAIN.MAC *
                   C    ;       ************
                   C    ;
                   C    ;----------------------------------
                   C    ;+ THIS IS THE INTERROGATOR MAIN PROGRAM +
                   C    ;----------------------------------
                   C    ;
                   C    ;Define the locations in RAM which hold the current
                   C    ;data address.
                   C    ;
FF0E               C    STRADD  EQU     GLOBAL+0EH      ;STORE ADDRESS POINTER
                   C    ;
                   C    ;If Q is set it is time to begin a measurement sequence.
                   C    ;
0F48'   C9 0FF7'   C    MAIN:   LBNQ    SHTDWN          ;SHUT DOWN IF NO Q
                   C    ;
                   C    ;This is the measurement sequence. Since it is approximately
                   C    ;one minute before the PING, enable interrupts to keep the
                   C    ;clock running and stop processing for one minute.
                   C    ;
0F4B'   7A         C    MSRSEQ: REQ                     ;INSURE THAT Q IS RESET
0F4C'   F8 17      C            LDI     LOW     (TICK)  ;RESET THE TICK FLAG
0F4E'   A7         C            PLO     R7
0F4F'   F8 00      C            LDI     00H
```

```
 0F51'   57             C           STR    R7
 0F52'   E3             C           SEX    R3
 0F53'   65             C           OUT    CLRINT           ;RESET INTERRUPT
 0F54'   00             C           DB     00H              ;HARDWARE AND INSURE
 0F55'   F8 62'         C           LDI    LOW     (INTRPT)
 0F57'   A1             C           PLO    R1               ;THAT R1 IS POINTING
 0F58'   F8 03'         C           LDI    HIGH    (INTRPT)
 0F5A'   B1             C           PHI    R1               ;AT INTERRUPT BEFORE
 0F5B'   70             C           RET                     ;INTERRUPTS ARE ENABLED.
 0F5C'   33             C           DB     33H
                        C           CALL   DELAY            ;CLOCK CAL. CONSTANT IS
 0F5D'   D4             C+
 0F5E'   00A2'          C+
 0F60'   067D           C           DW     067DH            ;200 mS. (OSC. START UP)
 0F62'   00             C           IDL                     ;WAIT ONE MINUTE
                        C     ;
                        C     ;Restore measurement interval counter to its original value
                        C     ;
 0F63'   F8 26          C     RSTMI: LDI   LOW     (MINOW)
 0F65'   A7             C           PLO    R7
 0F66'   F8 24          C           LDI    LOW     (HEXMI)
 0F68'   AA             C           PLO    RA               ;POINT AT MEAS. INT.
 0F69'   97             C           GHI    R7               ;USING RA A
 0F6A'   BA             C           PHI    RA               ;THE POINTER
 0F6B'   4A             C           LDA    RA               ;GET OLD VALUE
 0F6C'   57             C           STR    R7               ;AND DUPLICATE IT
 0F6D'   17             C           INC    R7               ;AT MINOW
 0F6E'   0A             C           LDN    RA
 0F6F'   57             C           STR    R7
                        C     ;
                        C     ;Convert current time to time code and store.
                        C     ;
 0F70'   F8 14          C           LDI    LOW     (HD+4)   ;POINT AT UNITS OF HOURS
 0F72'   A7             C           PLO    R7
 0F73'   07             C           LDN    R7               ;GET UNITS OF HOURS
 0F74'   AB             C           PLO    RB
                        C           CALL   RSB2A            ;SHIFT 4 BITS TO RA
 0F75'   D4             C+
 0F76'   00B7'          C+
 0F78'   04             C           DB     04H
 0F79'   27             C           DEC    R7               ;GET TENS OF HOURS
 0F7A'   07             C           LDN    R7
 0F7B'   AB             C           PLO    RB
                        C           CALL   RSB2A            ;SHIFT 2 BITS TO RA
 0F7C'   D4             C+
 0F7D'   00B7'          C+
 0F7F'   02             C           DB     02H
 0F80'   27             C           DEC    R7               ;GET UNITS OF DAYS
 0F81'   07             C           LDN    R7
 0F82'   AB             C           PLO    RB
                        C           CALL   RSB2A            ;SHIFT 4 BITS TO RA
 0F83'   D4             C+
 0F84'   00B7'          C+
 0F86'   04             C           DB     04H
 0F87'   27             C           DEC    R7               ;GET TENS OF DAYS
 0F88'   07             C           LDN    R7
```

```
0F89'   AB          C               PLO     RB
                    C               CALL    RSB2A           ;SHIFT 4 BITS TO RA
0F8A'   D4          C+
0F8B'   00B7'       C+
0F8D'   04          C               DB      04H
0F8E'   27          C               DEC     R7              ;GET HUNDREDS OF DAYS
0F8F'   07          C               LDN     R7
0F90'   AB          C               PLO     RB
                    C               CALL    RSB2A           ;SHIFT 2 BITS TO RA
0F91'   D4          C+
0F92'   00B7'       C+
0F94'   02          C               DB      02H
0F95'   8A          C               GLO     RA              ;GET LOW BYTE
0F96'   AF          C               PLO     RF              ;SAVE IT
0F97'   9A          C               GHI     RA              ;GET HI BYTE
0F98'   BF          C               PHI     RF              ;SAVE IT
                    C       ;
                    C       ;This is the measurement sequence. RA, RB, and RC are
                    C       ;used as travel time counters for F1, F2, and F3. RD is
                    C       ;used as a time out counter. The measurement sequence will
                    C       ;terminate when a reply from all three transponders has been
                    C       ;received, or RD rolls over to 0000. Since the counters are
                    C       ;incremented at a 4 kHz rate, the maximum measurement time
                    C       ;will not exceed 16.4 seconds.
                    C       ;
0F99'   F8 00       C       SNDPNG: LDI     00              ;RESET ALL COUNTERS
0F9B'   AA          C               PLO     RA
0F9C'   BA          C               PHI     RA
0F9D'   AB          C               PLO     RB
0F9E'   BB          C               PHI     RB
0F9F'   AC          C               PLO     RC
0FA0'   BC          C               PHI     RC
0FA1'   AD          C               PLO     RD
0FA2'   BD          C               PHI     RD
0FA3'   C4          C               NOP
0FA4'   C4          C               NOP                     ;MOVE PROGRAM POINTER TO
0FA5'   C4          C               NOP                     ;TOP OF LAST PAGE.
0FA6'   C4          C               NOP
                    C       ;
                    C       ;Wait for the leading edge of the 4 kHz timing signal.
                    C       ;
0FA7'   3F A7'      C       WAIT0:  BN4     WAIT0
0FA9'   37 A9'      C       WAIT1:  B4      WAIT1
0FAB'   E3          C               SEX     R3
0FAC'   63          C               OUT     PING            ;PING
0FAD'   00          C               DB      00H
                    C       ;
0FAE'   3F AE'      C       W0:     BN4     W0              ;WAIT FOR THE NEXT
0FB0'   37 B0'      C       W1:     B4      W1              ;RISING EDGE OF 4 KHZ
0FB2'   1D          C               INC     RD              ;COUNT IT
                    C       ;
                    C       ;Begin looking for reply to ping, and incrementing counters
                    C       ;if the reply is not detected.
                    C       ;
0FB3'   34 B6'      C               B1      TEST2           ;INCREMENT COUNTER IF
0FB5'   1A          C               INC     RA              ;NO RECEPTION, OTHERWISE
```

```
 0FB6'    35 B9'        C       TEST2:  B2      TEST3           ;SKIP TO NEXT TEST
 0FB8'    1B            C               INC     RB
 0FB9'    36 BC'        C       TEST3:  B3      TESTRD
 0FBB'    1C            C               INC     RC
 0FBC'    8D            C       TESTRD: GLO     RD              ;IF RD IS NOT ZERO
 0FBD'    CA 0FAE'      C               LBNZ    W0              ;CONTINUE TESTING
 0FC0'    9D            C               GHI     RD              ;AND INCREMENTING
 0FC1'    CA 0FAE'      C               LBNZ    W0              ;OTHERWISE, STORE DATA
                        C       ;
 0FC4'    E7            C       SAVIT:  SEX     R7              ;USE R7 AS THE POINTER
 0FC5'    F8 0E         C               LDI     LOW     (STRADD)
 0FC7'    A7            C               PLO     R7              ;GET CURRENT DATA
 0FC8'    47            C               LDA     R7              ;ADDRESS
 0FC9'    BD            C               PHI     RD              ;TRANSFER TO RD
 0FCA'    FB FF         C               XRI     0FFH            ;IF INTO GLOBAL PAGE
 0FCC'    C2 0FEF'      C               LBZ     ALSTOP          ;CEASE MEASUREMENTS
 0FCF'    07            C               LDN     R7              ;OTHERWISE, CONTINUE
 0FD0'    AD            C               PLO     RD
 0FD1'    9F            C               GHI     RF              ;STORE TIME
 0FD2'    5D            C               STR     RD
 0FD3'    1D            C               INC     RD
 0FD4'    8F            C               GLO     RF
 0FD5'    5D            C               STR     RD
 0FD6'    1D            C               INC     RD
 0FD7'    9A            C               GHI     RA              ;STORE TRAVEL TIME A
 0FD8'    5D            C               STR     RD
 0FD9'    1D            C               INC     RD
 0FDA'    8A            C               GLO     RA
 0FDB'    5D            C               STR     RD
 0FDC'    1D            C               INC     RD
 0FDD'    9B            C               GHI     RB              ;STORE TRAVEL TIME B
 0FDE'    5D            C               STR     RD
 0FDF'    1D            C               INC     RD
 0FE0'    8B            C               GLO     RB
 0FE1'    5D            C               STR     RD
 0FE2'    1D            C               INC     RD
 0FE3'    9C            C               GHI     RC              ;STORE TRAVEL TIME C
 0FE4'    5D            C               STR     RD
 0FE5'    1D            C               INC     RD
 0FE6'    8C            C               GLO     RC
 0FE7'    5D            C               STR     RD
 0FE8'    1D            C               INC     RD
 0FE9'    8D            C               GLO     RD              ;SAVE CURRENT STORE ADDRESS
 0FEA'    73            C               STXD
 0FEB'    9D            C               GHI     RD
 0FEC'    57            C               STR     R7
 0FED'    30 F7'        C               BR      SHTDWN
                        C       ;
 0FEF'    F8 43         C       ALSTOP: LDI     LOW     (GOFLG) ;SINCE THE CURRENT
 0FF1'    A7            C               PLO     R7              ;ADDRESS IS WITHIN
 0FF2'    F8 00         C               LDI     00H             ;GLOBAL PAGE, RESET THE
 0FF4'    57            C               STR     R7              ;GO FLAG BOTH
 0FF5'    17            C               INC     R7              ;HIGH AND LOW
 0FF6'    57            C               STR     R7              ;AND SHUT DOWN
                        C       ;
 0FF7'    7B            C       SHTDWN: SEQ                     ;LOCK POWER ON
 0FF8'   E3            C     PDLOOP: SEX     R3          ;RESET POWER CONTROL
 0FF9'   62            C             OUT     PWRRST      ;FLIP FLOP
 0FFA'   00            C             DB      00H
 0FFB'   71            C             DIS                 ;DISABLE INTERRUPTS
 0FFC'   33            C             DB      33H
 0FFD'   7A            C             REQ                 ;TURN POWER OFF AND
 0FFE'   00            C             IDL                 ;WAIT TILL IT DROPS
                       C             END
```

```
INTERROGATOR CONTROL/DATA LOGGER (PNAVLGR.MAC)  MACRO-18 3.36          PAGE    S
WOODS HOLE OCEANOGRAPHIC INST.  OCEAN ENGINEERING

MACROS:
CALL    CHAR?   ERROR?  GETFLG  RETURN  TYPMSG  WORD?

SYMBOLS:
ACTIVE  04C4'   AD10    0D5E'   AD100   0D67'   ADBIAS  0097'
ADDRS?  07BC'   ADONE   0094'   AERROR  0081'   AIHM    FF4A
ALSTOP  0FEF'   ARMIDL  056D'   ASHD    FF38    ASKOK   0113'
ASTH    FF3C    ASTK    0594'   ASTM    FF3F    ASUM    FF3E
AT      0457'   AT?     0BC2'   ATOH    0058I'  BADCHR  0108'
BCDHEX  0D4E'   BEL     0007    BHDONE  0D7A'   BYTOUT  0978'
CALCRC  0253'   CALL    003AI'  CALLCC  0A87'   CLEAR   03F7'
CLKTIC  03A4'   CLOOP   0A7B'   CLOSE   0911'   CLRCLO  0219'
CLRINT  0005    CMDIN   07D9'   CMPXIT  0144'   COLSET  0A2E'
COMPAR  012B'   CONFIG  0012    CORM    080D'   CPYTIM  02F2'
CR      000D    CRC     0A58'   CRCHI   FF0B    CRCLO   FF0C
CROOUT  0AA4'   CRLF    03D0'   CRLFSP  03D6'   CRTST   0A06'
CTST    012F'   DADONE  016C'   DATA    0006    DATAIN  0162'
DEARM   0DCB'   DECC    00AE'   DELAY   00A2'   DEVICE  07C9'
DIFFER  0143'   DIHM    FF45    DLE     0417'   DSHD    FF30
DSHFT   09EB'   DSUM    FF36    DTOA    0095'   ENTINT  037A'
EOL     03CF'   EQS     03F3'   ERROR   043A'   ERROUT  08F9'
ERVEC   0939'   ETX     0003    EXASK   012A'   EXCON   038F'
EXDLY   00B6'   EXINT   0360'   EXIT    03C4'   EXITC   0039'
EXITR   004A'   EXPHXN  0211'   FROM    03DD'   GET2HX  023CI'
GETCHR  0171'   GETDEC  019E'   GETHEX  016DI'  GFTOO   0F11'
GLOBAL  FF00    GOFLG   FF43    HD      FF10    HDONE   0077'
HELP    059E'   HEXMI   FF24    HLPOUT  0905'   HTOA    0085I'
I?      0888'   IDENT   08D2'   IDLE1   04FE'   INCHAR  0145I'
INCTH   034A'   INDEC   0196'   ING     041B'   INIT    0000'
INMINT  0D2A'   INTIM   0B9C'   INTRPT  0362'   ITYPE   00D7I'
LDADD   09AF'   LDREGS  0C27'   LDSCED  0CBE'   LDTIM   0B90'
LETST   098A'   LF      000A    LOAD    09A4'   LOCK    040D'
LTOP    0C26'   LYPYR1  02CA'   LYPYR?  02C5'   M1CLK   0282I'
M2CLK   0286'   MAIN    0F48'   MEAINT  048E'   MIMIN   0535'
MINOW   FF26    MINREM  04D1'   MODFLG  0A39'   MORL    082F'
MOVE    0ADF'   MOVIT   0B26'   MSRSEQ  0F4B'   NAME    03CC'
NCYCLE  0C5C'   NDA     014A'   NO      03FE'   NOCMD   08AE'
NOCR    0A4A'   NOLF    0A51'   NORUN   08C6'   NOTACT  0516'
NOTARM  0582'   NUL     0000    NXTADD  0D6D'   NXTD    09D2'
NXTLOC  0C72'   NXTM    FF18    NXTXOR  0331'   OK      041F'
OK?     0423'   ONNO0   0C3F'   ONNO1   0C44'   ONNO2   0C4A'
ONNO3   0C4F'   ONNO4   0C54'   OPEN    0925'   OVE     0444'
OVER    03EB'   P?      08B1'   PDLOOP  0FF8'   PHXIN   01E6I'
PING    0003    PN?     089B'   PNTR    0504'   PRMOUT  08ED'
PRMPT   0433'   PROMPT  003A    PS1HD   0353'   PWRRST  0002
QRYSCE  0DD6'   QUERRY  094D'   QUETIM  0B35'   RAM     1000
RAMTST  0BD5'   RAND    0C35'   RC$     03DA'   READY   0448'
RESTR   039A'   RETURN  004BI'  RMTST   0596'   RSB2A   00B7'
RSPEC   0BE9'   RSTFLG  091A'   RSTMI   0F63'   RSTRX   038A'
RTOP    0C34'   RUN     0AC0'   S1HD    FF29    S1UM    FF2F
S?      0875'   SAIL    03B0'   SALTTY  00CBI'  SARMI   0F05'
SAT     058C'   SAVIT   0FC4'   SAYIDL  0EED'   SAYNOT  0F2B'
SCDMSG  04B3'   SCED    045D'   SCRACH  FF05    SECS    044F'
SELECT  0001    SETAD   0D9A'   SGOFLG  0340'   SHFTC   01B5'
SHIFT   0078'   SHIFTC  0184'   SHRB    00B9'   SHTDWN  0FF7'
SIZE    F000    SNARM   0EF9'   SNDPNG  0F99'   SP      03D4'
SPACE   0020    SPEC    0AEA'   SPOUT   096F'   SPSP    03D3'
STACK   FFFF    STATUS  0007    STDAY   0465'   STHOUR  0478'
STMIN   0482'   STOP    007E    STORE   0A23'   STPCLK  0004
STRADD  FF0E    STRDEC  01D0'   STRNEW  034E'   T?      0862'
```

```
TEST2   0FB6'   TEST3   0FB9'   TESTRD  0FBC'   THRE?   00DC'
TICK    FF17    TIME    0459'   TO      03E5'   TSTDA   0154'
TSTGF   0300'   TSTHIC  00A6'   TSTHR   00E9'   TSTICK  0358'
TSTIME  0328'   TSTQ    02E4'   TSTSP   099C'   TXIT    010B'
TXMIT   0F37'   TYPADD  095D'   TYPEC   021FI'  U?      084F'
UM      FF16    UNLOCK  0411'   UPDATE  02B6'   VERCYC  0C6B'
VERIFY  0C6A'   W0      0FAE'   W1      0FB0'   WAIT0   0FA7'
WAIT1   0FA9'   WRITE   0C5D'   WRTERR  0C8D'   WST5    00B2'
X2HEX   0252'   XGETH   0195'   XINDEC  01C9'   XJNE    01C6'
XINTF   038E'
```

NO FATAL ERROR(S)

DOCUMENT LIBRARY

January 17, 1990

Distribution List for Technical Report Exchange

Attn: Stella Sanchez-Wade
Documents Section
Scripps Institution of Oceanography
Library, Mail Code C-075C
La Jolla, CA 92093

Hancock Library of Biology &
Oceanography
Alan Hancock Laboratory
University of Southern California
University Park
Los Angeles, CA 90089-0371

Gifts & Exchanges
Library
Bedford Institute of Oceanography
P.O. Box 1006
Dartmouth, NS, B2Y 4A2, CANADA

Office of the International
Ice Patrol
c/o Coast Guard R & D Center
Avery Point
Groton, CT 06340

NOAA/EDIS Miami Library Center
4301 Rickenbacker Causeway
Miami, FL 33149

Library
Skidaway Institute of Oceanography
P.O. Box 13687
Savannah, GA 31416

Institute of Geophysics
University of Hawaii
Library Room 252
2525 Correa Road
Honolulu, HI 96822

Marine Resources Information Center
Building E38-320
MIT
Cambridge, MA 02139

Library
Lamont-Doherty Geological
Observatory
Columbia University
Palisades, NY 10964

Library
Serials Department
Oregon State University
Corvallis, OR 97331

Pell Marine Science Library
University of Rhode Island
Narragansett Bay Campus
Narragansett, RI 02882

Working Collection
Texas A&M University
Dept. of Oceanography
College Station, TX 77843

Library
Virginia Institute of Marine Science
Gloucester Point, VA 23062

Fisheries-Oceanography Library
151 Oceanography Teaching Bldg.
University of Washington
Seattle, WA 98195

Library
R.S.M.A.S.
University of Miami
4600 Rickenbacker Causeway
Miami, FL 33149

Maury Oceanographic Library
Naval Oceanographic Office
Stennis Space Center
NSTL, MS 39522-5001

Marine Sciences Collection
Mayaguez Campus Library
University of Puerto Rico
Mayagues, Puerto Rico 00708

Library
Institute of Oceanographic Sciences
Deacon Laboratory
Wormley, Godalming
Surrey GU8 5UB
UNITED KINGDOM

The Librarian
CSIRO Marine Laboratories
G.P.O. Box 1538
Hobart, Tasmania
AUSTRALIA 7001

Library
Proudman Oceanographic Laboratory
Bidston Observatory
Birkenhead
Merseyside L43 7 RA
UNITED KINGDOM

50272-101

REPORT DOCUMENTATION PAGE	1. REPORT NO. WHOI 90-39	2.	3. Recipient's Accession No.

4. Title and Subtitle

A SAIL Compatible Three Channel Acoustic Navigation Interrogator

5. Report Date

September 1990

6.

7. Author(s)

Stephen P. Liberatore

8. Performing Organization Rept. No.

WHOI 90-39

9. Performing Organization Name and Address

Woods Hole Oceanographic Institution
Woods Hole, Massachusetts 02543

10. Project/Task/Work Unit No.

11. Contract(C) or Grant(G) No.

(C) N00014-82-C-0152
(G) N00014-85-C-0379

12. Sponsoring Organization Name and Address

Office of Naval Research

13. Type of Report & Period Covered

Technical Report

14.

15. Supplementary Notes

This report should be cited as: Woods Hole Oceanographic Inst. Tech. Rept., WHOI 90-39.

16. Abstract (Limit: 200 words)

Ocean Acoustic Tomography data are significantly degraded if mooring motion is unknown. An autonomous instrument employing a solid state data logger designed to track and record mooring motion is described.

Navigation is accomplished by simultaneously interrogating each of three bottom mounted transponders positioned in an equilateral triangle around the mooring's anchor at a range approximately equal to the depth of the tracked instrument. The three round-trip travel times thus obtained having a resolution of 125uS and a SNR dependent jitter of less than 1.5mS, define a unique instrument position and are recorded along with the time of day and day of year.

The measurement period, the system clock and the program start time are set via a 20mA SAIL. Since the standby power requirement is negligible compared to the battery capacity, the instrument may be programmed months in advance of the deployment.

System endurance varies with the measurement period, however, typical programs permit navigation for up to 21 months at 12 points per day.

Upon recovery, the navigator data may be down-loaded via SAIL directly to the storage medium of a suitable computer.

17. Document Analysis a. Descriptors

navigation
autonomous
mooring

b. Identifiers/Open-Ended Terms

c. COSATI Field/Group

18. Availability Statement Approved for public release; distribution unlimited.	19. Security Class (This Report)	21. No. of Pages 104
	20. Security Class (This Page)	22. Price